A three-dimensional nonsingular simulation of rigid manipulators.

Verbos, Robert M.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A THREE-DIMENSIONAL NONSINGULAR SIMULATION OF RIGID MANIPULATORS

by

Robert M. Verbos

September 1988

Thesis Advisor: D. Smith

Approved for public release; distribution is unlimited.

Unclassified

Security Classification of this page

REPORT DOCUMENTATION PAGE

1a Report Security Classification: Unclassified		1b Restrictive Markings	
2a Security Classification Authority		3 Distribution Availability of Report: Approved for public release; distribution is unlimited.	
2b Declassification/Downgrading Schedule			
4 Performing Organization Report Number(s)		5 Monitoring Organization Report Number(s)	
6a Name of Performing Organization: Naval Postgraduate School	6b Office Symbol (*If Applicable*): 69	7a Name of Monitoring Organization: Naval Postgraduate School	
6c Address (*city, state, and ZIP code*): Monterey, CA 93943-5000		7b Address (*city, state, and ZIP code*): Monterey, CA 93943-5000	
8a Name of Funding/Sponsoring Organization	8b Office Symbol (*If Applicable*)	9 Procurement Instrument Identification Number	
8c Address (*city, state, and ZIP code*)		10 Source of Funding Numbers	
		Program Element Number	Project No / Task No / Work Unit Accession No

11 Title (*Include Security Classification*): A Three-Dimensional Nonsingular Simulation of Rigid Manipulators

12 Personal Author(s): Robert M. Verbos

13a Type of Report	13b Time Covered	14 Date of Report (*year, month,day*)	15 Page Count
Master's Thesis	From To	September 1988	140

16 Supplementary Notation: The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 Cosati Codes			18 Subject Terms (*continue on reverse if necessary and identify by block number*)
Field	Group	Subgroup	Robot, Simulation, Validation, Verification

19 Abstract (*continue on reverse if necessary and identify by block number*)

Robot manipulators have been studied, using various approaches to obtain the kinematic and dynamic equations which describe their motion. Conventional body-oriented robot arm kinematic equations have the disadvantage that a singular condition occurs when two successive links of the manipulator are aligned. When this occurs, the Jacobian matrix which relates the end effector motion to the joint angle variations becomes singular and is not invertible, resulting in motion that can not be simulated. This thesis extends the previous work done in the investigation of a nonsingular Newton Euler approach to forward dynamic equations interpreted in a global (inertia) fixed reference frame. Specifically, the previous results are extended into validation of the approach for three-dimensional motion, including gravitational effects. In addition, the comparison and verification of the motion of an actual robotic manipulator to the simulation is investigated.

20 Distribution/Availability of Abstract	21 Abstract Security Classification	
[X] unclassified/unlimited [ ] same as report [ ] DTIC users	Unclassified	
22a Name of Responsible Individual: Associate Professor D. Smith	22b Telephone (*Include Area code*): (408) 646-3383	22c Office Symbol: 69Sm

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted security classification of this page

All other editions are obsolete Unclassified

Approved for public release; distribution is unlimited.

A Three-Dimensional Nonsingular Simulation of Rigid Manipulators

by

Robert M. Verbos
Lieutenant, United States Navy
B.E.E., Villanova University, 1981

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MECHANICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1988

ABSTRACT

Robot manipulators have been studied, using various approaches to obtain the kinematic and dynamic equations which describe their motion. Conventional body-oriented robot arm kinematic equations have the disadvantage that a singular condition occurs when two successive links of the manipulator are aligned. When this occurs, the jacobian matrix which relates the end effector motion to the joint angle variations becomes singular and is not invertible, resulting in motion that can not be simulated. This thesis extends the previous work done in the investigation of a nonsingular Newton Euler approach to forward dynamic equations interpreted in a global (inertia) fixed reference frame. Specifically, the previous results are extended into validation of the approach for three-dimensional motion, including gravitational effects. In addition, the comparison and verification of the motion of an actual robotic manipulator to the simulation is investigated.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

A	Sine Wave Input Torque Amplitude
AA	Acceleration of Point A
AB	Acceleration of Point B
AG1	The Acceleration Vector of the Center of Gravity for Link 1
AG2	Same as AG1 but for Link 2
AG3	Same as AG1 but for Link 3
AMP	Magnitude of the Unit Vector of the Directional Cosines
AX0	Linear Acceleration of Joint 0 in the X Direction
AY0	Linear Acceleration of Joint 0 in the Y Direction
AZ0	Linear Acceleration of Joint 0 in the Z Direction
AX1, AY1, AZ1	Linear Acceleration of Link 1 in the Respective Direction
AX2, AY2, AZ2	Linear Acceleration of Link 2 in the Respective Direction
AX3, AY3, AZ3	Linear Acceleration of Link 3 in the Respective Direction
CPROD	Subroutine to Calculate a Cross Product of Two Vectors
DEGRA	DSL Conversion From Degrees to Radians

DRCANX(1), DRCANX(2), DRCANX(3)	Direction Cosine Angle in Degrees in the Fixed Coordinate System From the X Axis for Links 1-3, Respectively
DRCANY(1), DRCANY(2), DRCANY(3)	Same as above except from the Y Axis
DRCANZ(1), DRCANZ(2), DRCANZ(3)	Same as above except from the Z Axis
DRCRAX(1) DRCRAX(2) DRCRAX(3)	Direction Cosine Angle in Radians in the Fixed Coordinate System from the X Axis for Links 1-3, Respectively
DRCRAY(1), DRCRAY(2), DRCRAY(3)	Same as above except from the Y Axis
DRCRAZ(1), DRCRAZ(2), DRCRAZ(3)	Same as above except from the Z Axis
DRCSX(1), DRCSX(2), DRCSX(3)	The Argument of the Direction Cosine Angle in the X Direction for Links 1-3, Respectively
DRCSY(1), DRCSY(2), DRCSY(3)	Same as above except in the Y Direction
DRCSZ(1), DRCSZ(2), DRCSZ(3)	Same as above except in the Z Direction
DQ	A 27 x 1 Column Matrix Obtained by Multiplying MATA by MATB
FX0	Computed Force in the X Direction at Joint 0
FY0	Computed Force in the Y Direction at Joint 0
FZ0	Computed Force in the Z Direction at Joint 0
FX1, FY1, FZ1	Same as above except at Joint 1

FX2, FY2, FZ2	Same as above except at Joint 2
G	Gravitational Constant
I	Counter
IA	Row Dimension of MATA and MATB Used in IMSL Routine LEQT2F
IER	Error Parameter Used in IMSL Routine LEQT2F
IGDT	Accuracy Test Used in IMSL Routine LEQT2F
IMAT	A 3 x 3 x 3 Matrix which contains the Global Inertias of Links 1, 2, and 3
IXX	A 3 x 2 Matrix of Moment of Inertia for the Two-Element Composite Body of Links 1-3 About the X Axis
IYY	Same as IXX but About the Y Axis
IZZ	Same as IXX but About the Z Axis
IXZ	A 3 x 2 Matrix of Product of Inertia for the Two-Element Composite Body of Links 1-3 About the XZ Coordinate Axis
IXY	Same as IXZ but for the XY Axis
IYZ	Same as IXZ but for the YZ Axis
IXXT	Total Moment of Inertia of Links 1-3 About the X Axis
IYYT	Same as IXXT but About Y Axis
IZZT	Same as IXXT but About Z Axis
IXZT	Same as IXXT but About XZ Axis
IXYT	Same as IXXT but About XY Axis
IYZT	Same as IXXT but About YZ Axis
JX0	Location of Joint 0 in the X Direction

JY0	Location of Joint 0 in the Y Direction
JZ0	Location of Joint 0 in the Z Direction
JX1, JY1, JZ1	Same as above except for Joint 1
JX2, JY2, JZ2	Same as above except for Joint 2
L	A 3 x 2 Matrix, Distance from Center of Gravity of the Link to the Center of Mass at the Link End
LCOGX	A 1 x 3 Matrix, Location of the Link Center of Gravity in the X Direction
LCOGY	Same as LCOGX but for the Y Direction
LCOGZ	Same as LCOGX but for the Z Direction
LEVELQ	Error Set for Routine LEQT2F
LEVLDQ	Error Set for Routine LEQT2F
LIMAT	A 3 x 3 x 3 Matrix which Contains the Local Inertias of Links 1, 2, and 3
LNCOG1	The Length of the Vector from Joint 0 to the COG of Link 1
LNCOG2	The Length of the Vector from Joint 1 to the COG of Link 2
LNCOG3	The Length of the Vector from Joint 2 to the COG of Link 3
M	Number of the Right-Hand Side Used in LEQT2F
MASS	A 3 x 2 Matrix of the Masses of the Elements of the Link Composite
M1	Total Mass of Link 1
M2	Total Mass of Link 2
M3	Total Mass of Link 3

MATA	A 27 x 27 Matrix Consisting of the Coefficients of the Unknown Variables
MATB	A 27 x 1 Vector Consisting of the Coefficients of Knowns on Input, and the Solution from the IMSL Routine LEQT2F on Output
MATC	A 27 x 1 Column Vector which Contains the Results of the Multiplication MATA * MATD
MATD	A 27 x 1 Column Vector which Contains the Knowns in the Simulation Validation
MI	X Component, Cross-Product from Subroutine CPROD
MJ	Y Component, Cross-Product from Subroutine CPROD
MK	Z Component, Cross-Product from Subroutine CPROD
MI1, MJ1, MK1	Cross-Product of WD x R used in the Validation Program in the the Calculation of Acceleration
MI2, MJ2, MK2	Cross Product of W x W x R used in the Validation Program to Calculate Acceleration
MIC0, MJC0, MKC0	Cross-Product of W1 x W1 x RBG1 in the X, Y, and Z Direction
MIC1, MJC1, MKC1	Cross-Product of W1 x W1 x RAG1 in the X, Y, and Z Direction
MIC2, MJC2, MKC2	Cross-Product of W2 x W2 x RBG2 in the X, Y, and Z Direction
MIC3, MJC3, MKC3	Cross-Product of W2 x W2 x RAG2 in the X, Y and Z Direction
MIC4, MJC4, MKC4	Cross-Product of W3 x W3 x RBG3 in the X, Y, and Z Direction
N	Order of MATA and Number of Rows in MATB
P	Phase Angle of Sin Wave
RADEG	DSL Conversion from Radians to Degrees

RX	A 3 x 2 Matrix of the Distance from the COG of the Link to the Center of the Link Masses in the X Direction
RY	Same as RX except in Y direction
RZ	Same as RX except in Z direction
RAG1	A 1 x 3 Vector of the Distance of Point A to the COG of Link 1 in the X, Y, and Z Direction
RAG2	Same as RAG1 except for Link 2
RAG3	Same as RAG1 except for Link 3
RBG1	A 1 x 3 Vector of the Distance of Point B to the COG of Link 1 in the X, Y, and Z Direction
RBG2	Same as RBG1 except for Link 2
RBG3	Same as RBG2 except for Link 3
T0X, T0Y, T0Z	Torque at Joint 0 in the X, Y, and Z Direction
T1FNC, T2FNC	Torque at Joints 1-2 beyond the Gravitational Torque
T1X, T1Y, T1Z	Torque at Joint 1 in the X, Y, and Z Direction
T2X, T2Y, T2Z	Torque at Joint 2 in the X, Y, and Z Direction
TG1, TG2	Gravitation Torque at Joints 1-2
TIPX, TIPY, TIPZ	Position of the End Effector
TMAT	A 3 x 3 Transformation Matrix which Transforms the Local Coordinates to Global
TMATTR	A 3 x 3 Matrix which is the Transpose of TMAT
TMPMAT	A Temporary 3 x 3 Matrix used in Matrix Multiplication
VECTA, VECTB	A 1 x 3 Vector Used in the Subroutine CPROD
W	Frequency of Sine Wave

WG1, WG2, WG3	Weight of Links 1-3
W1, W2, W3	A 1 x 3 Vector of the Angular Velocity of Links 1-3
WDX, WDY, WDZ	Angular Acceleration of Links 1-3 in the X, Y, and Z Direction
WKAREA	Work Area used in the IMSL Routine LEQT2F
X1, X2, X3	Initial Location of the COG of Link 1-3 in the X Direction
Y1, Y2, Y3	Initial Location of the COG of Link 1-3 in the Y Direction
Z1, Z2, Z3	Initial Location of the COG of Link 1-3 in the Z Direction

ACKNOWLEDGMENTS

I would like to thank Professor Smith for his guidance and patience in this difficult endeavor, Professor Chang for his assistance inthe development of some of the dynamics, and Mr. Tom Christian of the Mechanical Engineering Department technical staff for his many hours of help in rebuilding the robot. I also thank my wife, Vickie, for her understanding during the past six months.

I. INTRODUCTION

The study of robotics is a fairly new area in the fields of science and engineering which has fundamentals that can be traced back to many basic disciplines, such as electrical engineering, mechanical engineering, and computer science. The mechanical engineering interest in the study of robotic manipulators can be traced back to as early as the 1940s, although the first industrial robots were not introduced until the late 1950s and early 1960s [Ref. 1]. Mechanical engineers were mostly interested in two areas of robotics: the control of the linkage and the study of its motion. While these two areas are interrelated, this thesis focuses on the second: the study of the motion of the linkage.

The study of robot motion (dynamics) is further divided into the study of arm dynamics and robot arm kinematics. Both these divisions are subdivided into a direct and an inverse problem. The direct dynamics problem is the calculation of link state variables such as acceleration, velocity, and joint angles from the known joint torques. The inverse dynamic problem assumes state variables are known and the joint torques are to be calculated. In the kinematics, the direct problem is the determination of the end effector position and its orientation from a given set of link geometries and joint angles. The inverse problem is the calculation of the link geometries and joint angles for a given end effector position.

Various methods to solve the inverse kinematic problem use matrix transformations, inversion, and multiplication, and as such are subjected to singularity problems [Ref. 2]. Singularity occurs when two successive links are aligned (i.e., the angle between the links is 0 or 180 degrees). When a singularity occurs, the matrices involved are also singular and not invertible.

A method to avoid the singularity problem was developed at the Naval Postgraduate School using Newtonian dynamics in terms of a globally fixed coordinate system [Ref. 3]. This method treats each link as a free body with the forces and moments applied at the joints and the use of Newton's law to derive the equations of motion. In this thesis, the development of the free body analysis approach to rigid manipulators dynamics is continued and extended to three dimensions and including gravitational effects.

II. PROBLEM STATEMENT

Theoretical dynamic approaches are well developed and have been extensively used with simulations. However, in manipulator studies, these accepted methods all have a singularity that arises when two successive links are aligned. This does not allow the motion to be simulated [Ref. 3]. This shortfall cannot be tolerated in the modelling of an actual robot or robotic manipulator because it is of the utmost importance to accurately know the arm position at all times (for real-time control purposes) [Ref. 2]. Therefore, the problem directed by this thesis research project was:

> Continue the development of the Newton Euler free-body approach for a three-link manipulator extended to three-dimensional motion and including gravitational effects. In addition, compare simulation results with those of an actual robotic arm. [Ref. 4]

III. BACKGROUND

A. THEORY

The basic theory for the Newton Euler free-body analysis to overcome the singularity problems of conventional robot dynamics was developed by Sadrettin Altinok [Ref. 3]. The theory of treating each link of the manipulator as a free body with respect to a global coordinate system leads to the following 27 equations for a three-link manipulator.

$$FX0 + M1*AX1 - FX1 = 0 \tag{1}$$

$$FY0 + M1*AY1 - FY1 = 0 \tag{2}$$

$$FZ0 + M1*AZ1 - FZ1 = -WG1 \tag{3}$$

$$AX1 + RB1Z*WD1Y - RB1X*WD1Z = AX0 - MIC0 \tag{4}$$

where MIC0 = (X component of)

$$\underline{WD1} \times \underline{RBG1} + \underline{W1} \times (\underline{W1} \times \underline{RBG1}) \tag{4a}$$

$$AY1 - RB1Z*WD1X + RB1X*WD1Z = AY0 - MJC0 \tag{5}$$

where MJC0 = (Y component of equation 4a).

$$AZ1 + RBY1*WD1X - RB1Z*WD1Y = AZ0 - MKC0 \quad (6)$$

where MKC0 = (Z component of equation 4a).

$$RB1Z*FY0 - RB1Y*FZ0 - IXXT1*WD1X + IXYT1*WD1Y + IXZT1*WD1Z - RA1Z*FY1 + RA1Y*FZ1 = T1X - T0X \quad (7)$$

$$-RB1Z*FX0 + RB1X*FZ0 + IXYT1*WD1X - IYYT1*WD1Y + IYZT1*WD1Z + RA1Z*FX1 - RA1X*FZ1 = T1Y - T0Y \quad (8)$$

$$RB1Y*FX0 - RB1Z*FY0 + IXZT1*WD1X + IYZT1*WD1Y - IZZT1*WD1Z - RA1Y*FX1 + RA1X*FY1 = T1Z - T0Z \quad (9)$$

$$FX1 + M2*AX2 - FX2 = 0 \quad (10)$$

$$FY1 + M2*AY2 - FY2 = 0 \quad (11)$$

$$FZ1 + M2*AZ2 - FZ2 = -WG2 \quad (12)$$

$$-AX1 - RA1Z*WD1Y + RA1Y*WD1Z + AX2 + RB2Z*WD2Y - RB2Y*WD2Z = MIC1 - MIC2 \quad (13)$$

where MIC1 = (X component of)

$$\underline{WD1} \times \underline{RA1} + \underline{W1} \times (\underline{W1} \times \underline{RA1}) \quad (13a)$$

where MIC2 + (X component of)

$$\underline{WD2} \times \underline{RB2} + \underline{W2} \times (\underline{W2} \times \underline{RB2}) \qquad (13b)$$

$$-AY1 + RA1Z*WD1X - RA1X*WD1Z + AY2 - RB2Z*WD2X + RB2X*WD2Z = MJC1 - MJC2 \qquad (14)$$

where MJC1 = (Y component of equation 13a) and where MJC2 = (Y component of equation 13b).

$$-AZ1 - RA1Y*WD1X + RA1X*WD1Y + AZ2 + RB2Y*WD2X - RB2X*WD2Y = MKC1 - MKC2 \qquad (15)$$

where MKC1 = (Z component of equation 13a) and where MKC2 = (Z component of equation 13b).

$$RB2Z*FY1 - RB2Y*FZ1 - IXXT2*WD2X + IXYT2*WD2Y + IXZT2*WD2Z - RA2Z*FY2 + RA2Y*FZ2 = T2X - T1X \qquad (16)$$

$$-RB2Z*FX1 + RB2X*FZ1 + IXYT2*WD2X - IYYT2*WD2Y + IYZT2*WD2Z + RA2Z*FX2 - RA2X*FZ2 = T2Y - T1Y \qquad (17)$$

$$RB2Y*FX1 + RB2X*FY1 + IXZT2*WD2X + IYZT2*WD2Y - IZZT2*WD2Z - RA2Y*FX2 + RA2X*FY2 = T2Z - T1Z \qquad (18)$$

$$FX2 + M3*AX3 = 0 \quad (19)$$

$$FY2 + M3*AY3 = 0 \quad (20)$$

$$FZ2 + M3*AZ3 = -WG3 \quad (21)$$

$$-AX2 - RA2Z*WD2Y + RA2Y*WD2Z + AX3 + RB3Z*WD3Y - RB3Y*WD3Z = MIC3 - MIC4 \quad (22)$$

where MIC3 = (X component of)

$$\underline{WD2} \text{ X } \underline{RA2} + \underline{W2} \text{ X } (\underline{W2} \text{ X } \underline{RA2}) \quad (22a)$$

where MIC4 = (X component of)

$$\underline{WD3} \text{ X } \underline{RB3} + \underline{W3} \text{ X } (\underline{W3} \text{ X } \underline{RB3}) \quad (22b)$$

$$-AY2 + RA2Z*WD2X - RA2X*WD2Z + AY3 - RB3Z*WD3X + RB3X*WD3Z = MJC3 - MJC4 \quad (23)$$

where MJC3 = (Y component of equation 22a) and where MJC4 = (Y component of equation 22b).

$$-AZ2 - RA2Y*WD2X + RA2X*WD2Z + AZ3 + RB3Y*WD3X - RB3X*WD3Y = MKC3 - MKC4 \quad (24)$$

where MKC3 = (Z component of equation 22a) and where MKC4 = (Z component of equation 22b).

$$RB3Z*FY2 - RB3Y*FZ2 - IXXT3*WD3X + IXYT3*WD3Y + IXZT3*WD3Z = - T2X \quad (25)$$

$$-RB3Z*FX2 + RB3X*FZ2 + IXYT3*WD3X - IYYT3*WD3Y + IYZT3*WD3Z = - T2Y \quad (26)$$

$$RB3Y*FX2 - RB3X*FY2 + IXZT3*WD3X + IYZT3*WD3Y - IZZT3*WD3Z = - T2Z \quad (27)$$

B. COMPUTATION

The Dynamic Simulation Language (DSL) was used to simulate the direct dynamics problem. A matrix (MATA, a 27 x 27 matrix) was created from the known coefficients of the unknown variables in the previous 27 equations. This was possible due to the assumption that the change in the inertia of the links, the link velocities, and the joint position must be extremely small during a simulated step interval and can thus be treated as being constant during the step. In the direct dynamics problem, a vector (MatB, 27 x 1) was generated from the following components: joint torques, the centers of gravity of the link, and the calculated cross-products of angular velocities (Figure 1). This gave a system of equations in the form of equation 28:

Figure 1. **Matrix Entries for Simulation**

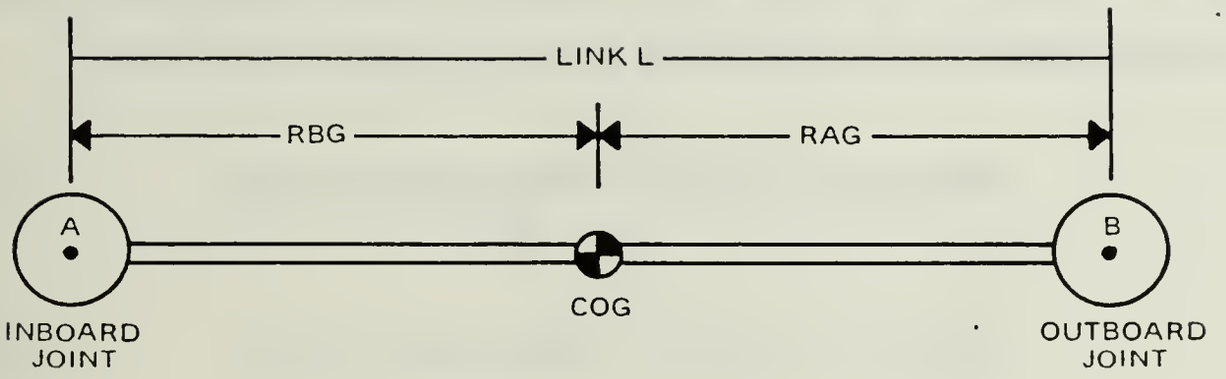
$$\underline{A} * \underline{x} = \underline{b} \qquad (28)$$

where A and b are known and x is an unknown vector (which contains the forces at the link ends and the angular and linear accelerations of each link). The IMSL routine LEQT2F (a linear equation solver) was used to obtained the vector x. [Ref. 3 and Ref. 5]

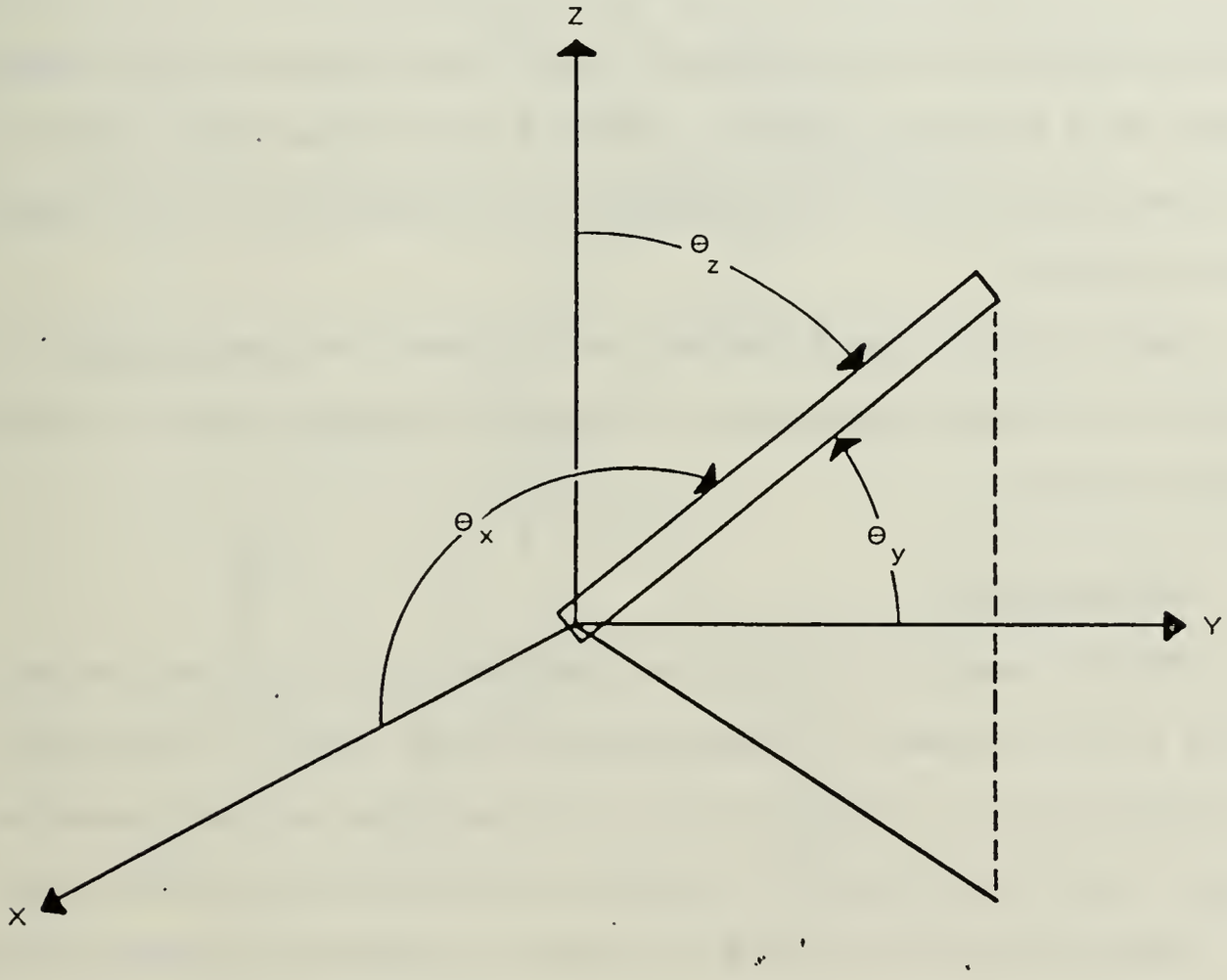
In the inverse problem, MATA was developed the same as in the direct problem as discussed above, but now vector x was known (from the history of position data) and vector b was found by straight matrix multiplication of equation 28. [Ref. 3 and Ref. 5]

The previous work used transformation matrix operations to obtain joint orientations [Ref. 3]. This was done because joint orientation cannot be obtained directly from the integration of rotational velocity [Ref. 6 and Ref. 7]. The approach used in this thesis was one of simple geometric relationships. It was reasoned that the linear accelerations were either known or calculated from the matrix operation of LEQT2F (equation 28). These linear accelerations were the linear accelerations of the link center of gravities with respect to the earth fixed (inertial) coordinate system which were integrated to obtain the linear velocities. The velocities were further integrated to obtain the position of the center of gravities.

Knowing the earth fixed coordinates of the centers of gravity, and that each link can be represented by a rigid body [Ref. 8], it was known that the center of gravity must be on the vector from the inboard joint to the end of the link, or next joint (Figure 2). Once the



(a) Center of Gravity



(b) Directional Cosine Angles

Figure 2. **Link Geometries**

vector was known to the center of gravity, the directional cosines were calculated from the following three equations for each link.

$$DRCOSX = [LCOGX - JX(inboard)]/LNCOG \quad (29)$$

$$DRCOSY = [LCOGY - JY(inboard)]/LNCOG \quad (30)$$

$$DRCOSZ = [LCOGZ - JZ(inboard)]/LNCOG \quad (31)$$

where LCOGX is the X location of the CG with respect to the inertia frame; JX is the X location of the inboard joint location with respect to the inertia; and LNCOG is magnitude of the length from the inboard joint to the CG.

Once the directional cosines were known, the same method as used by previous investigators was used to calculate the MATA and MATB matrices.

C. CONSTRAINTS

With the desire to simulate the motion of an actual robotic manipulator and to compare the simulation with data from the manipulator, constraints had to be added into the simulation so the simulation would take into account the the joint mechanical restrictions. In theory, each joint is a one degree of freedom connection. However, in the chosen arm, the zero joint rotates in the Z direction only, while joints one and two rotate about both the X and Y axes (Figure 3). The previously mentioned constraints led to the following equations:

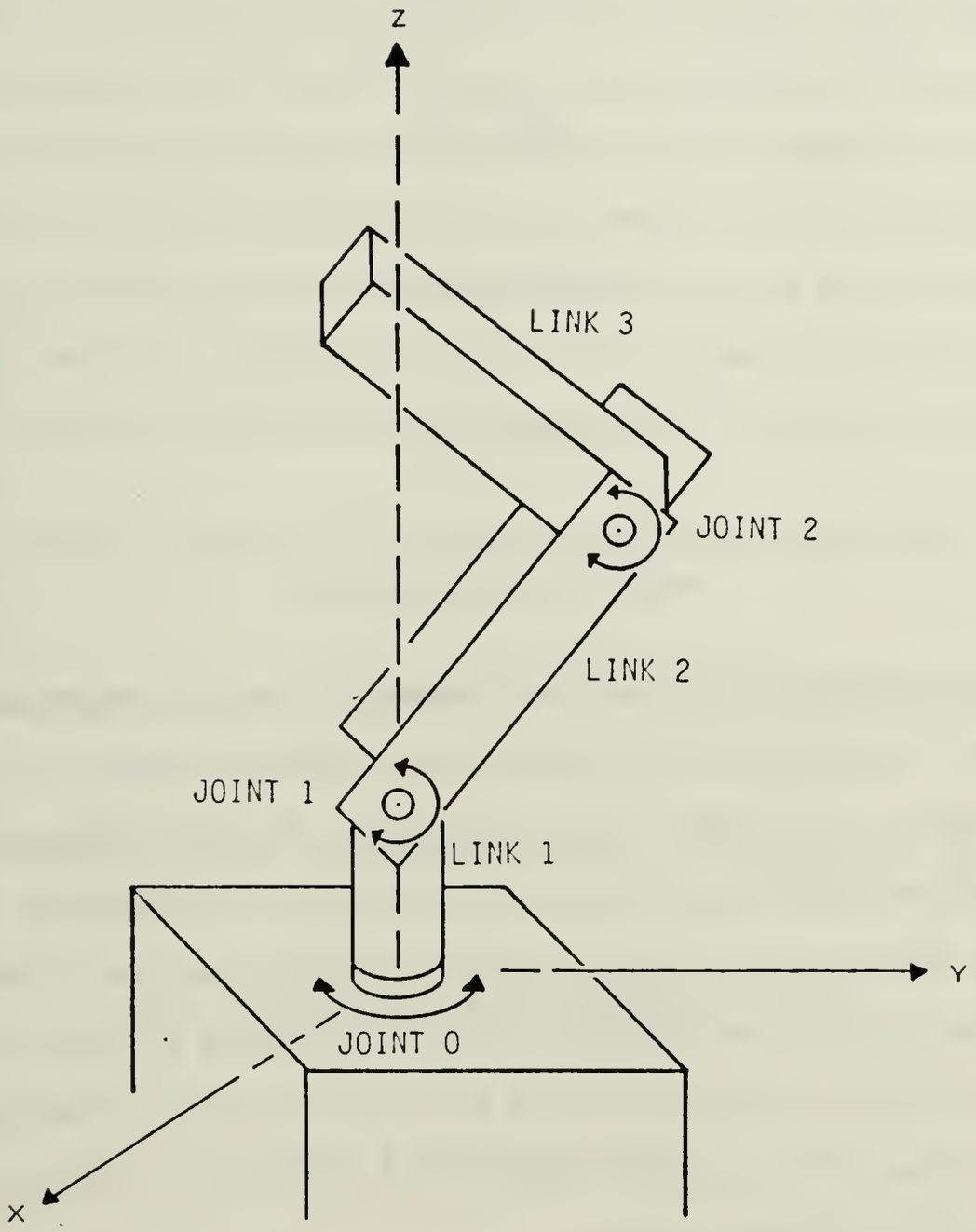


Figure 3. **Robot Axis Rotation with Constraints**

$$AX1 = 0 \tag{32}$$

$$AY1 = 0 \tag{33}$$

$$AZ1 = 0 \tag{34}$$

$$WDX1 = 0 \tag{35}$$

$$WDY1 = 0 \tag{36}$$

$$WDZ1 = WDZ2 = WDZ3 \tag{37}$$

The concept for entering the constraints into the simulation was developed in Reference 3 and led to the constraint section of the DSL code as seen in Appendix A and Appendix B. The basic concept which was developed was to zero out the row only in MATA, then set the diagonal of that row to one and the corresponding row of MATB to zero. This concept was changed to include zeroing out the row and column of MATA corresponding to the row of vector x, which was to be zero, along with setting the diagonal of the row to 1 and the corresponding row of MATB to zero.

D. GRAVITATIONAL TORQUES

The simulation of gravitational effects on the arm required that an equalizing torque be applied at the joints to overcome the gravitational

D. GRAVITATIONAL TORQUES

The simulation of gravitational effects on the arm required that an equalizing torque be applied at the joints to overcome the gravitational pull (Figure 4). This torque varied as the Z location of the center of gravity of links 2 and 3 varied because the magnitude of the moment arm varied as the link moved. The lever arm which affected the gravitational torque was the projection of the vector from the inboard joint to the link center of gravity onto the XY plane (Figure 5). Knowing this lever arm and the weight of the links, the following equations were used to calculate the equilibrium torque needed to overcome gravity:

$$MARM2A = SQRT (LCOGX2*LCOGX2 + LCOGY2*LCOGY2) \quad (38)$$

$$MARM2B = SQRT (LCOGX3*LCOGX3 + LCOGY3*LCOGY3) \quad (39)$$

$$MARM3 = SQRT ((LCOGX3-JX2)**2 + (LCOGY3-JY2)**2) \quad (40)$$

$$TG1 = MAR2A * WG2 + MARM2B * WG3 \quad (41)$$

$$TG2 = MARM3 * WG3 \quad (42)$$

The equations above represent the magnitude of the required torque which must be resolved into its X and Y components (Figure 6).

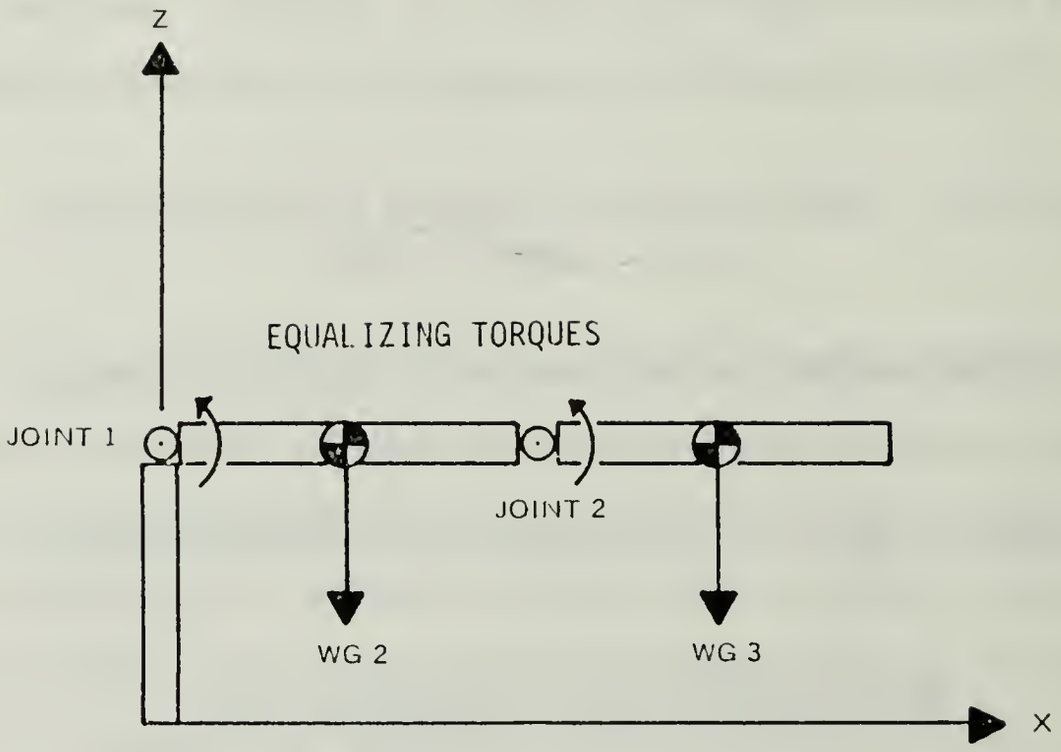


Figure 4. **Gravitational Torques**

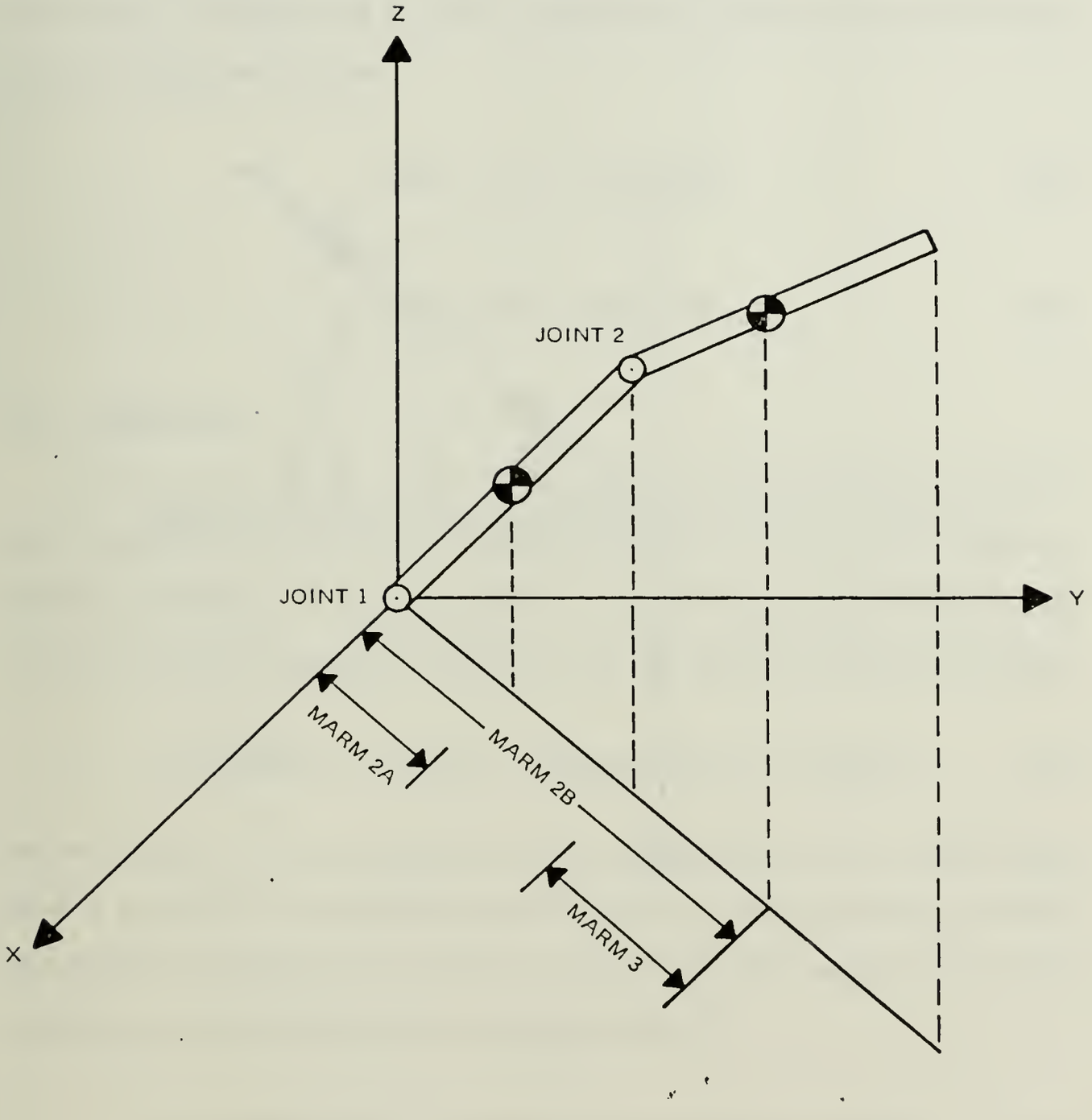


Figure 5. **Gravitational Torque Moment Arms**

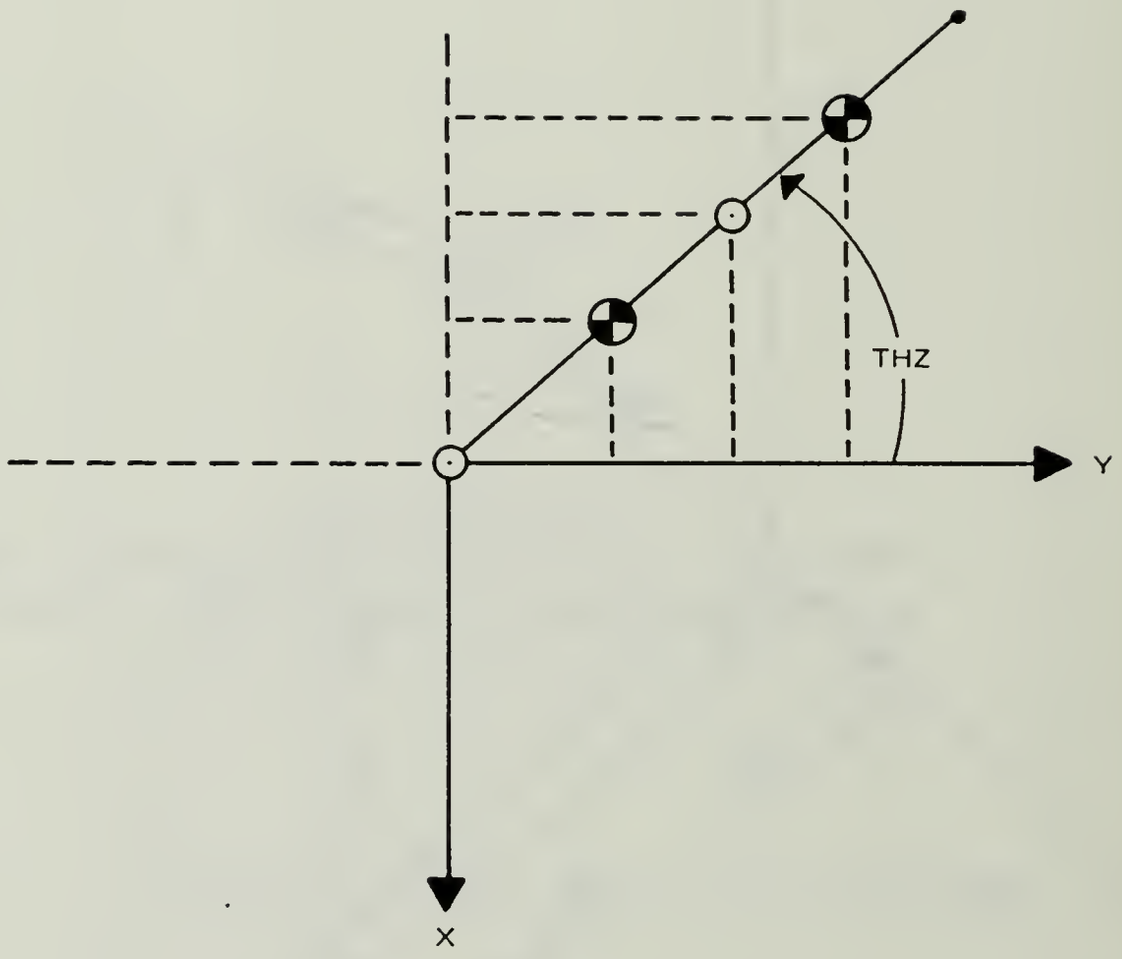


Figure 6. **Gravitational Torque X and Y Components**

With the given constraints, this was done in the following manner. Angle THZ was calculated from the second integral of WDZ1 (the base link only revolved around its vertical axis). Then its cosine and sine were used to obtain the X and Y components of the gravitational equilibrium torque as follows:

$$TGX = TG * COS (THZ) \quad (43)$$

$$TGY = TG * SIN (THZ) \quad (44)$$

E. INERTIAS

The inertias of the links of the robot were calculated the first time from the equations of Reference 2 in the global (inertial) reference frame and then transferred to the local coordinates (Figure 7) by the use of a transformation [Ref. 9], giving the following equation:

$$Inertia(local) = TMAT * Inertia(global) * TMATTR \quad (45)$$

where MATA is a transformation matrix based on the link angles [Ref. 1 and Ref. 10]. Knowing that the local inertias remain constant, the global inertias for each link were calculated for each time iteration by the use of the inverse transformation below:

$$Inertia(global) = TMATTR * Inertia(local) * TMAT \quad (46)$$

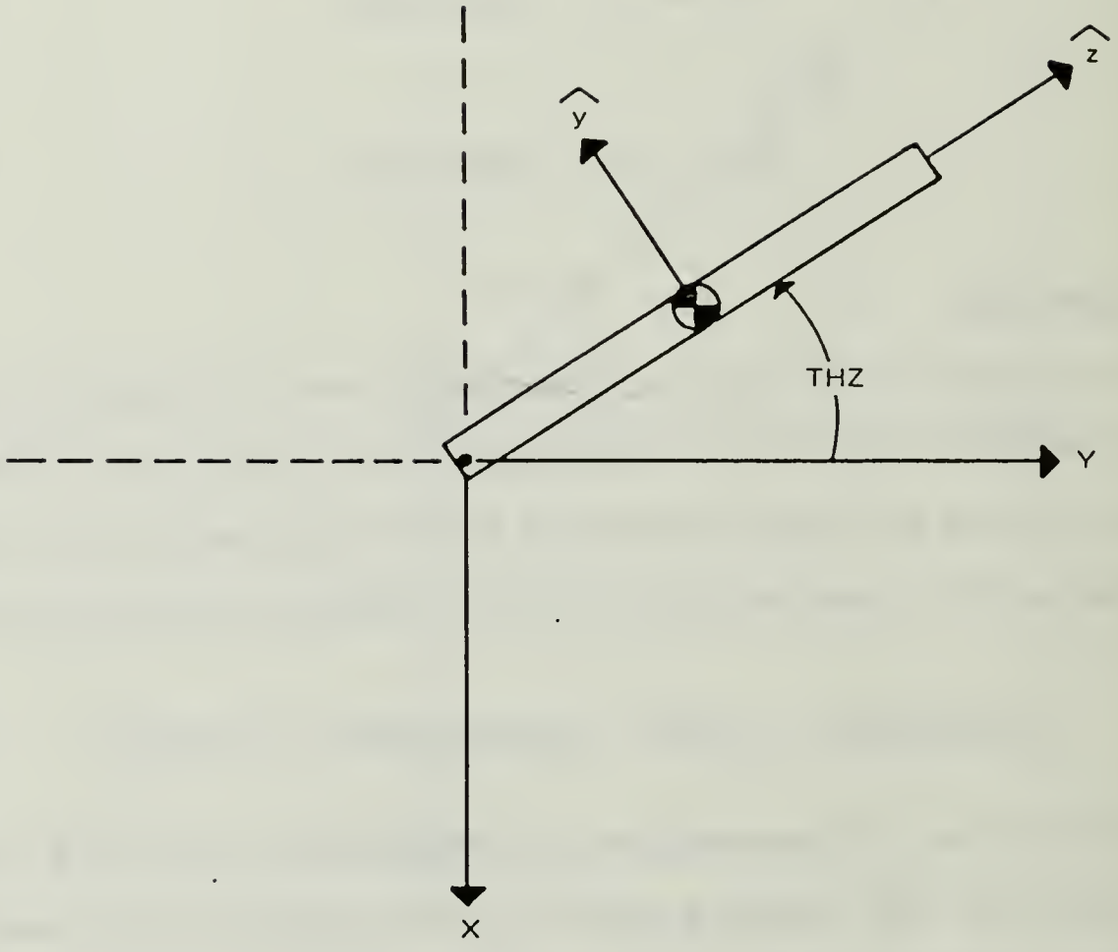


Figure 7. **Inertia Coordinate Systems**

IV. SIMULATION NONSINGULAR MOTION VALIDATION

A. TWO-DIMENSIONAL MOTION VALIDATION

A known motion which had adjacent links aligned was that of the two-dimensional double pendulum (Figure 8). A comparison of the simulation to the theoretical solution was used to validate the nonsingularity, the motion tracking, and the gravitational effects of the simulation.

The theoretical solution [Ref. 11] used a Lagrange equation which only included the linear kinetic energy of the system (equation 47), while the simulation included both the linear and rotational kinetic energies (equation 48). To account for the difference, the inertia in equation 48 was set to a small value. Note that it was not possible to set the inertias to zero because the inertia entries in matrix MATA were on the diagonal and would thus not have allowed the inverse to be calculated.

$$\text{Kinetic Energy} = 1/2 * m * v^2 \tag{47}$$

$$\text{Kinetic Energy} = 1/2 * m * v^2 + 1/2\, I * w^2 \tag{48}$$

The point mass method developed in Reference 3 allowed the first mass of the double pendulum to be set to a value equal to that of

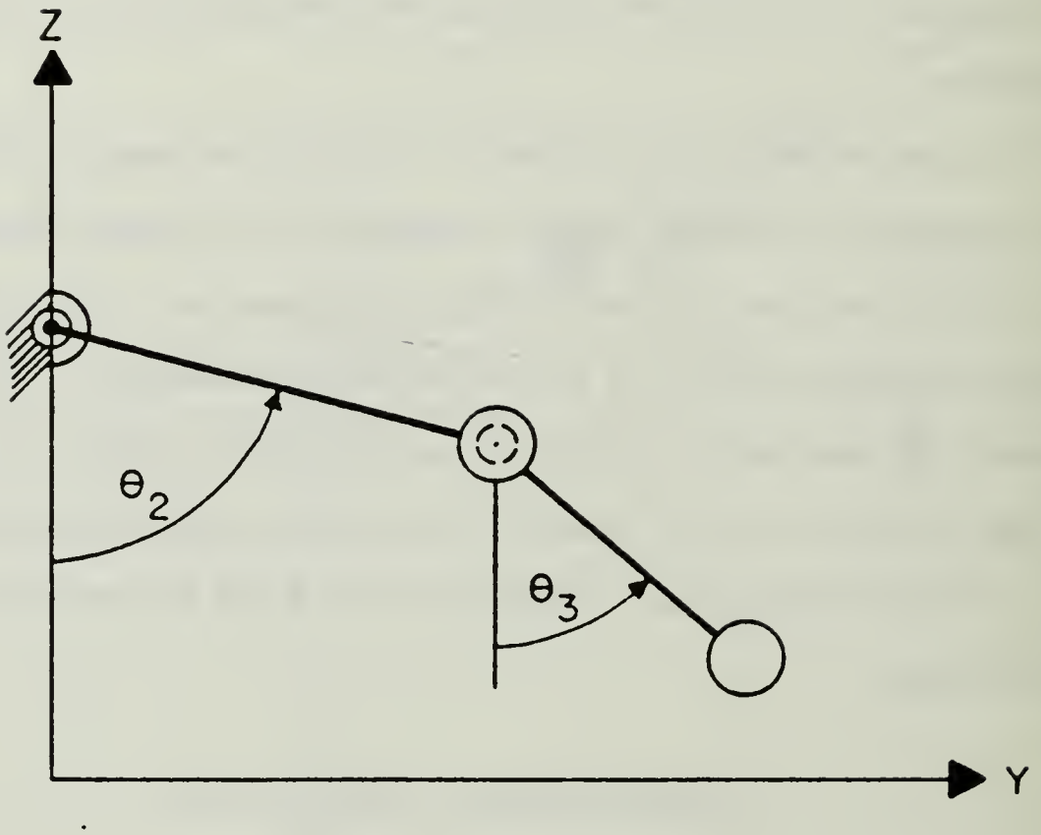


Figure 8. **Double Pendulum Configuration**

the outer point mass of link 2 plus the inner point mass of link 3 (equation 49). The second mass of the double pendulum was set to the value of the outer point mass of link 3 (equation 50).

$$m1\ (dp) = m\ (2,2) + m\ (3,1) \tag{49}$$

$$m2\ (dp) = m\ (3.2) \tag{50}$$

The lengths of the double pendulum were the same as the link lengths (Figure 8).

The results of the theoretical and simulation program for the double pendulum problem agreed (Figures 9 and 10), and would have been identical had the inertia been set to zero. Tracking of the motion through the potential singular points presented no difficulties to the simulation (Figure 11).

B. THREE-DIMENSIONAL NONSINGULAR MOTION

A motion which had potential singular points was chosen which validated the nonsingularity of the simulation in three dimensions (Figure 12). The motion which was chosen was to input the joint 1 and joint 2 angles as a time function and allow the third joint angle to remain zero, so links 2 and 3 were always aligned and potentially singular. This chosen motion also had all three links aligned at time equal to 0.785 seconds (Figure 12).

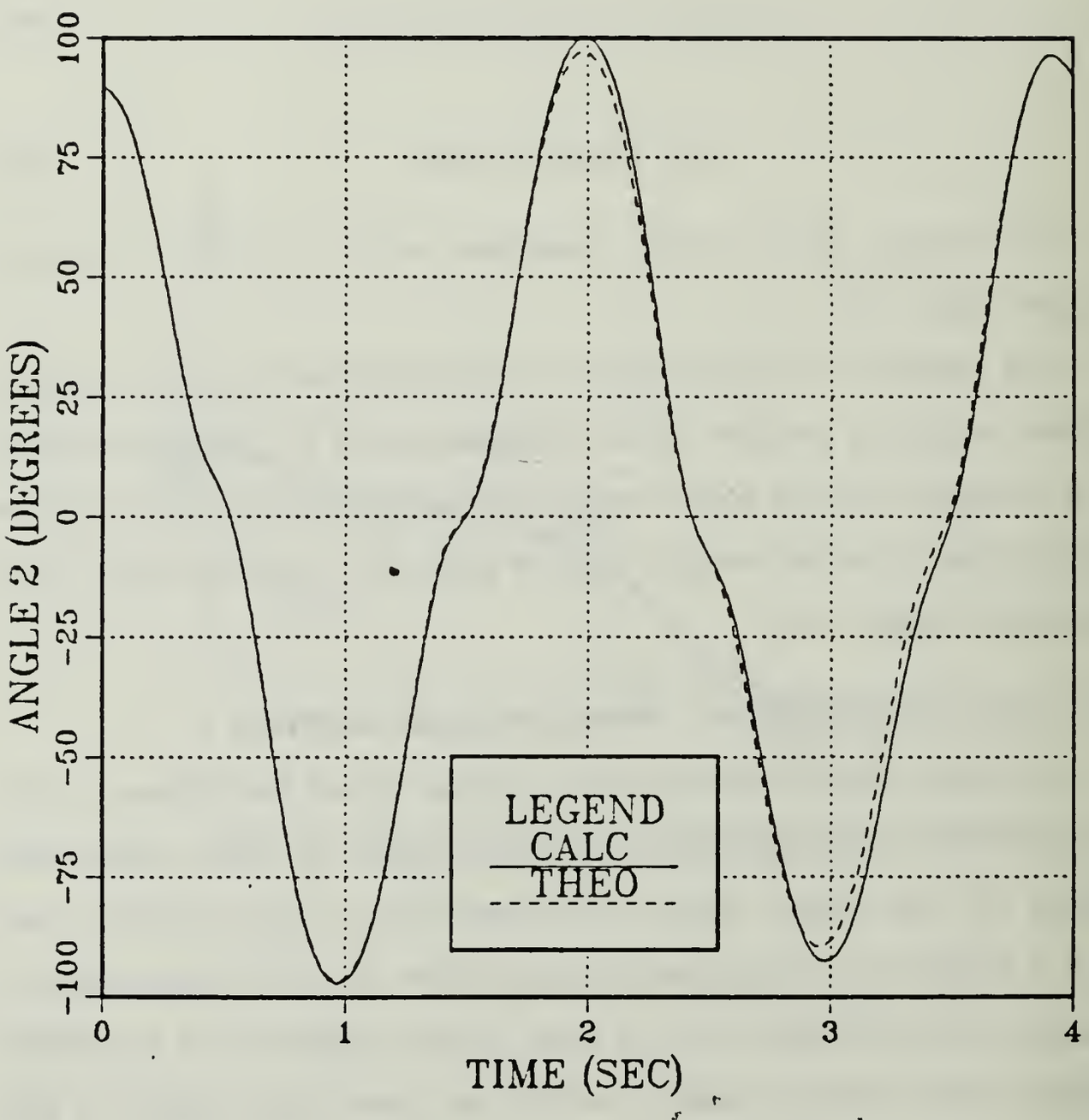


Figure 9. **Double Pendulum Angle 2**

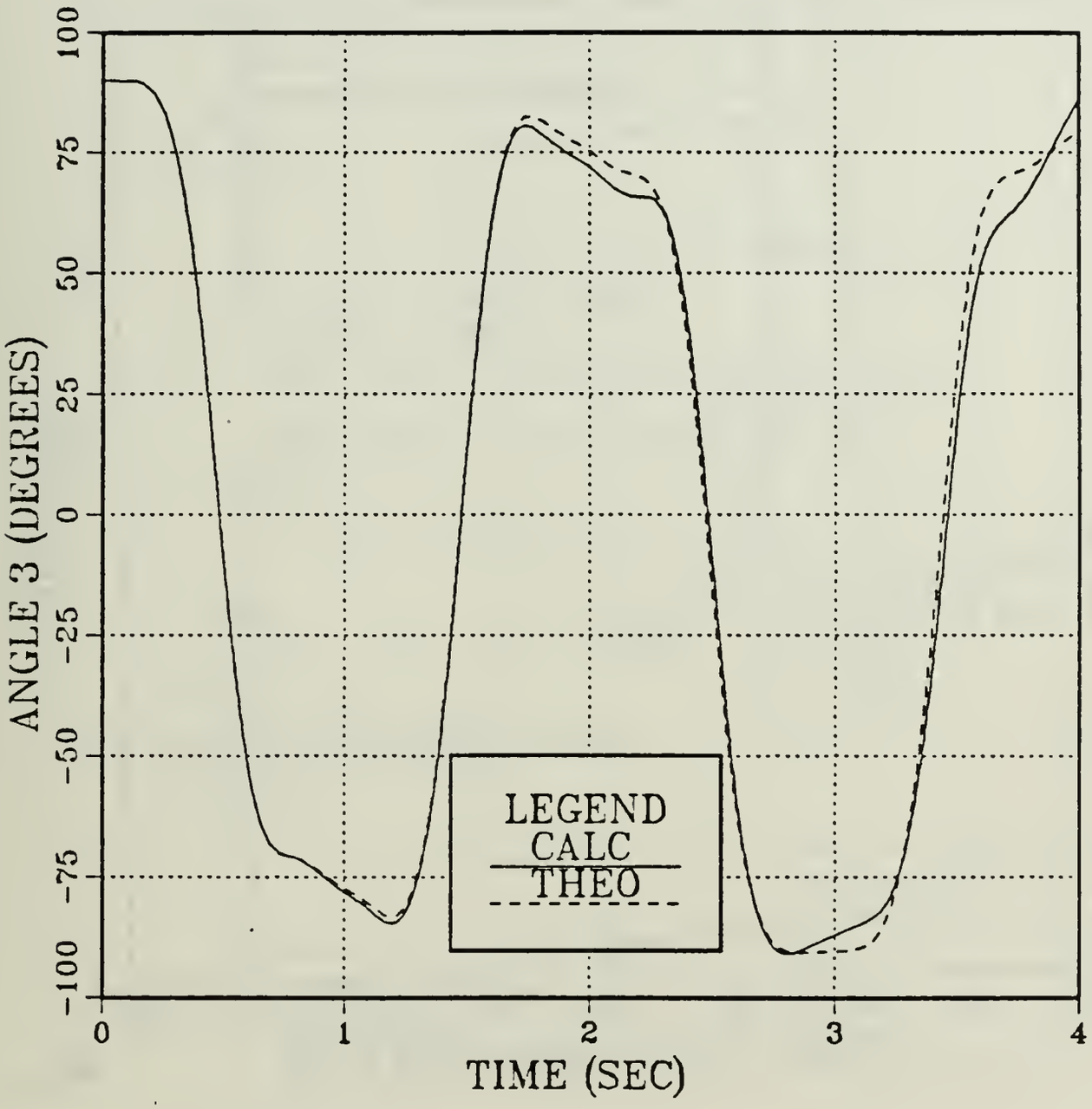


Figure 10. **Double Pendulum Angle 3**

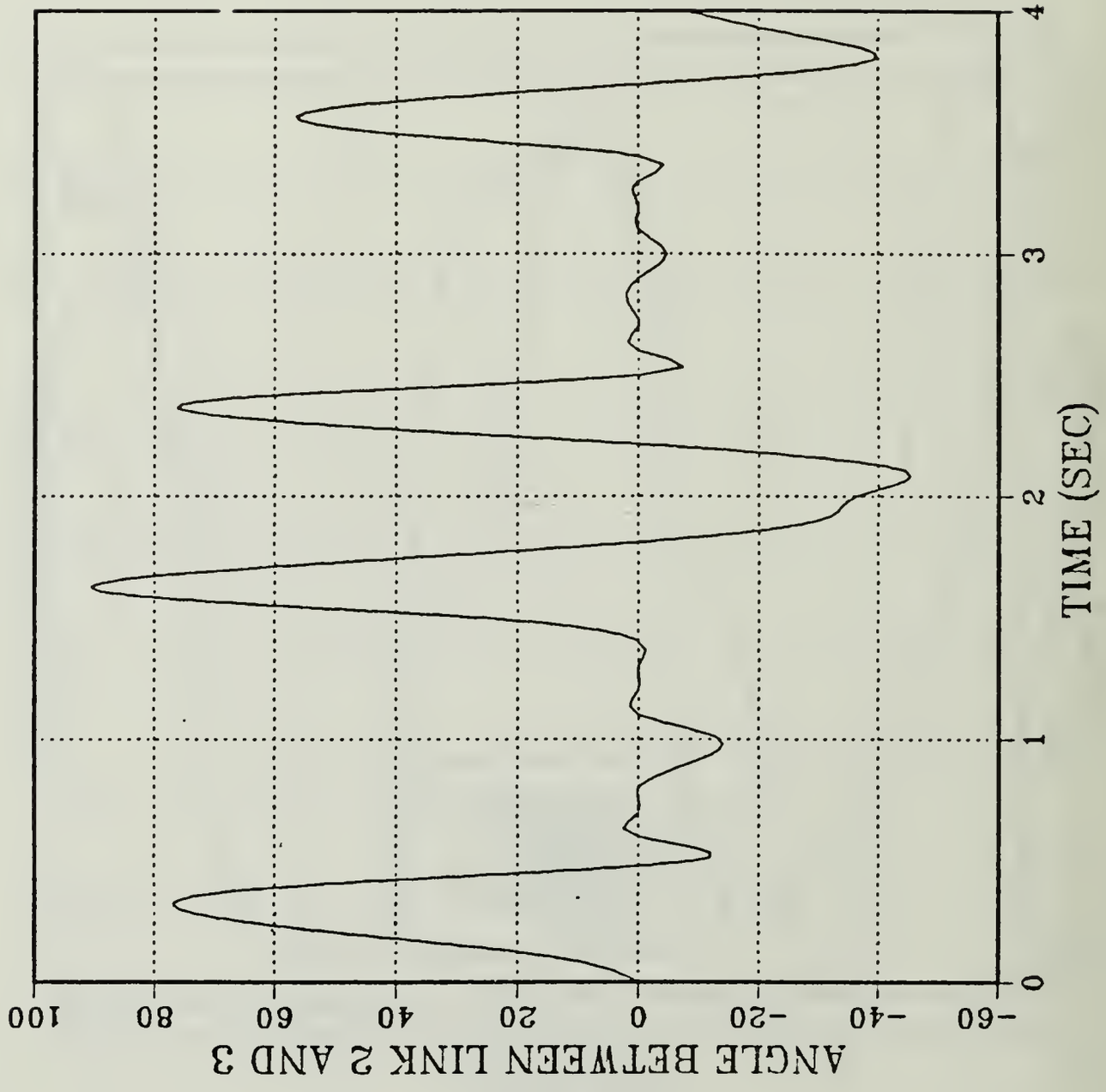


Figure 11. **Double Pendulum Simulation Through Potential Singular Points (angle = 0)**

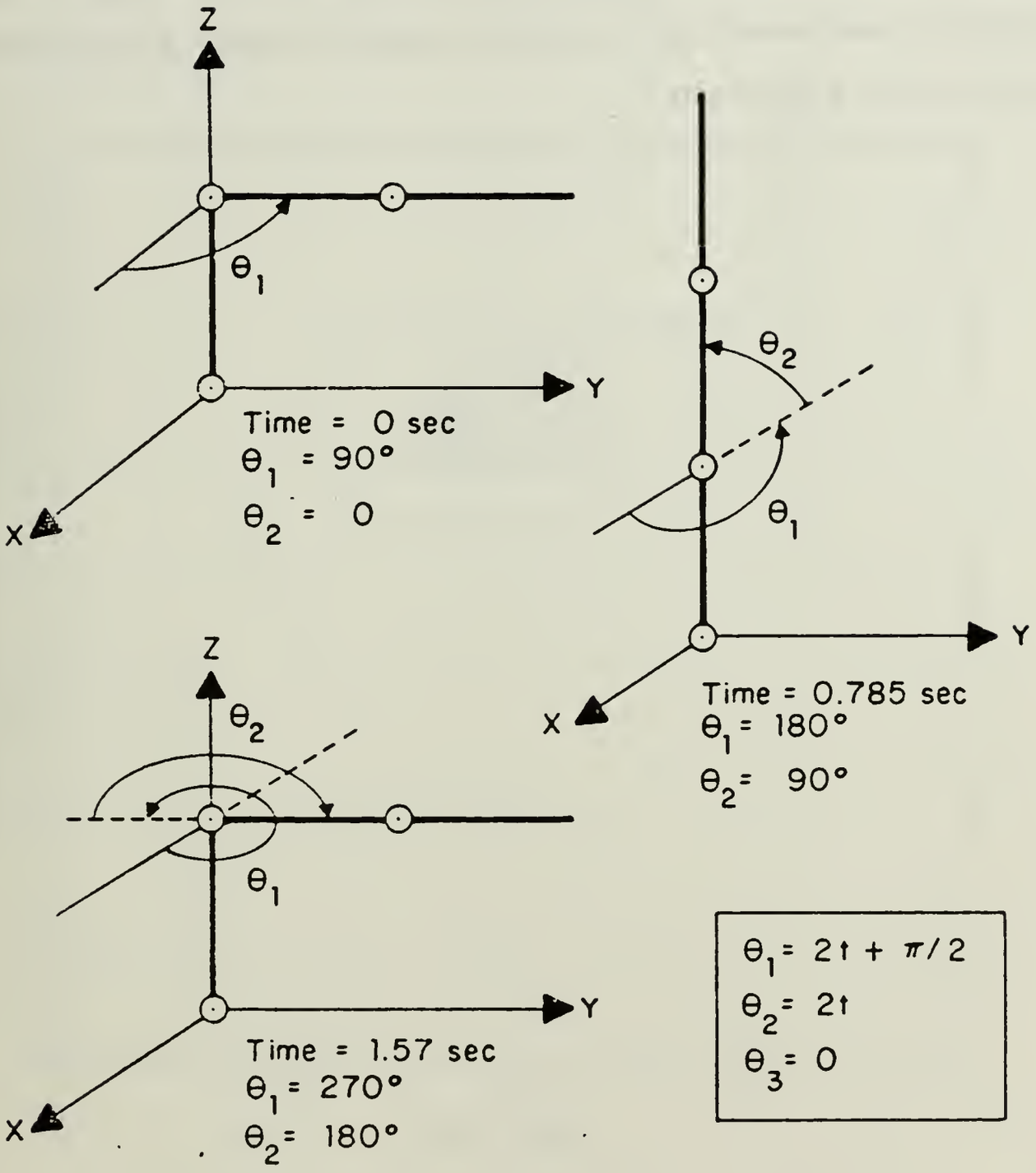


Figure 12. **Three-Dimensional Nonsingular Validation**

The simulation again had no difficulties tracking the input motion through the potentially singular points (Figure 13). The error in the predicted and simulated end effector position (Figure 14) was never greater than 1.0 percent.

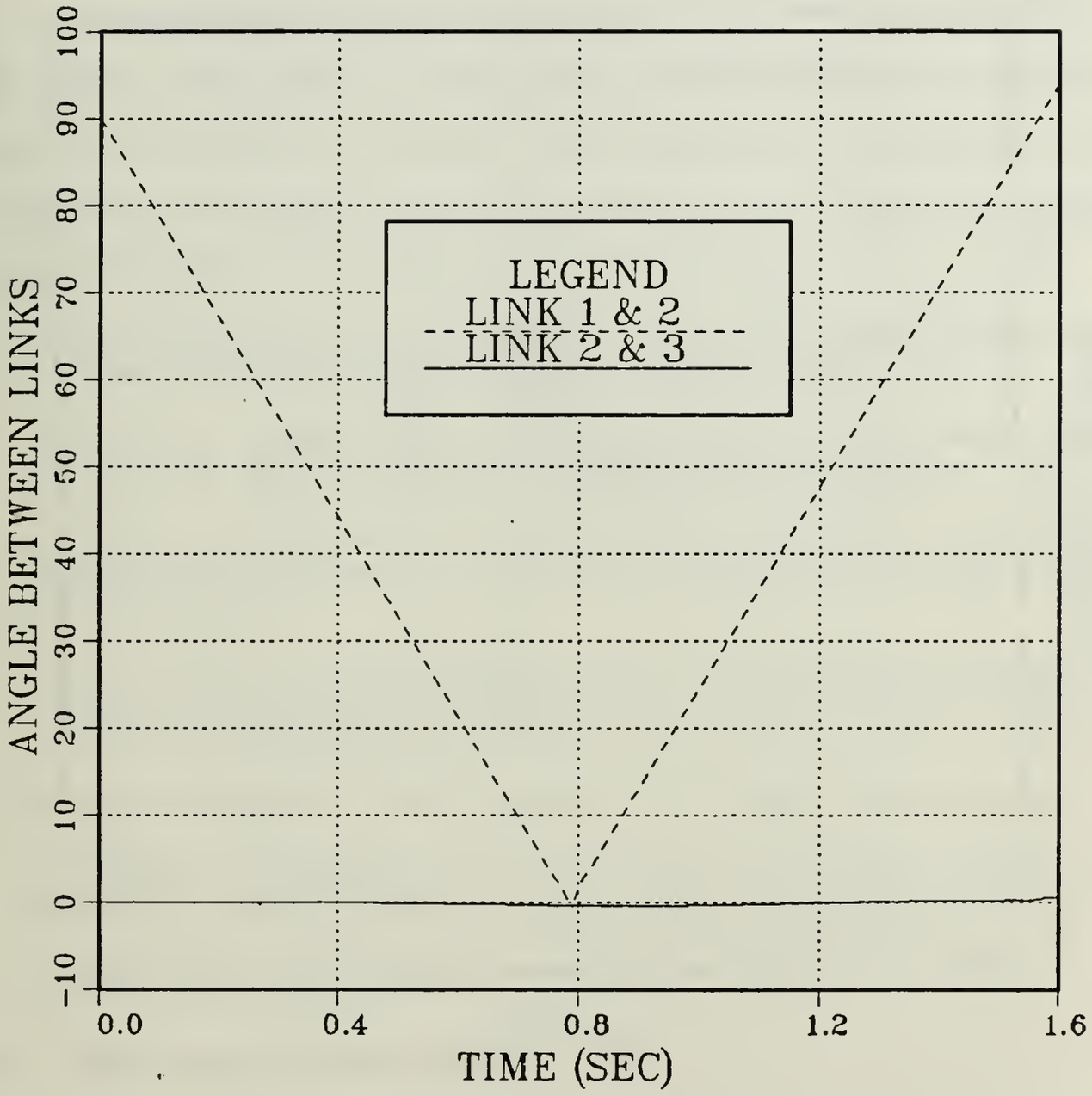


Figure 13. **Three-Dimensional Potential Singular Points**

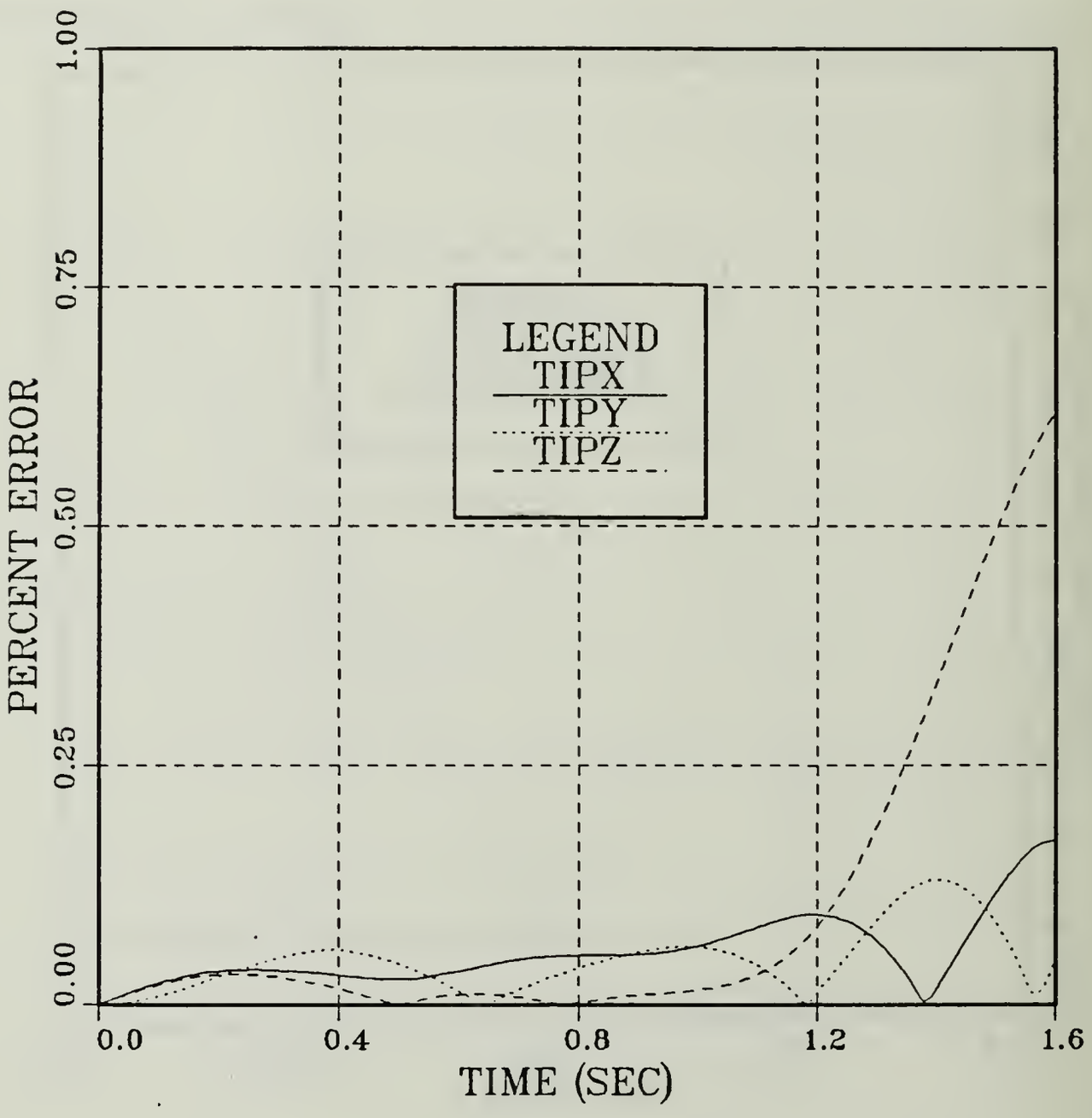


Figure 14. **End Effector Position Error**

V. EXPERIMENTAL VALIDATION

A. MANIPULATOR

A robot manipulator was chosen to check the agreement of the simulation to data from an actual robot arm. The arm that was available was the NEPTUNE II (Figure 15), which is a low-performance, hydraulic-operated arm. This arm required certain hardware modifications, namely:

- the remounting of the electronic solenoid boards to facilitate access and provide protection from hydraulic leaks;
- installation of bleed holes in the major joint pistons to allow air to be bled off when the system has been opened for repairs;
- design and manufacture of a pump and accumulator stand that would provide spray protection for the manipulator in the case of malfunctions;
- modification to the plumbing of the pump and accumulator to include an electrically controlled solenoid valve operated from the Electronic Control Panel (ECP);
- rewiring the pump to be electrically controlled from the ECP;
- redesign and manufacture of the waist joint (joint 0) shaft and coupling, along with that of several piston end plates; and
- an overhaul of the complete electronics.

B. PRESSURE TRANSDUCERS

Pressure transducers which had previously been installed on the NEPTUNE II arm at joints 1 and 2 (Figure 16) allowed for data collection to validate the two-dimensional simulation with gravitational effects. Initially, the pressure transducers had to be calibrated.

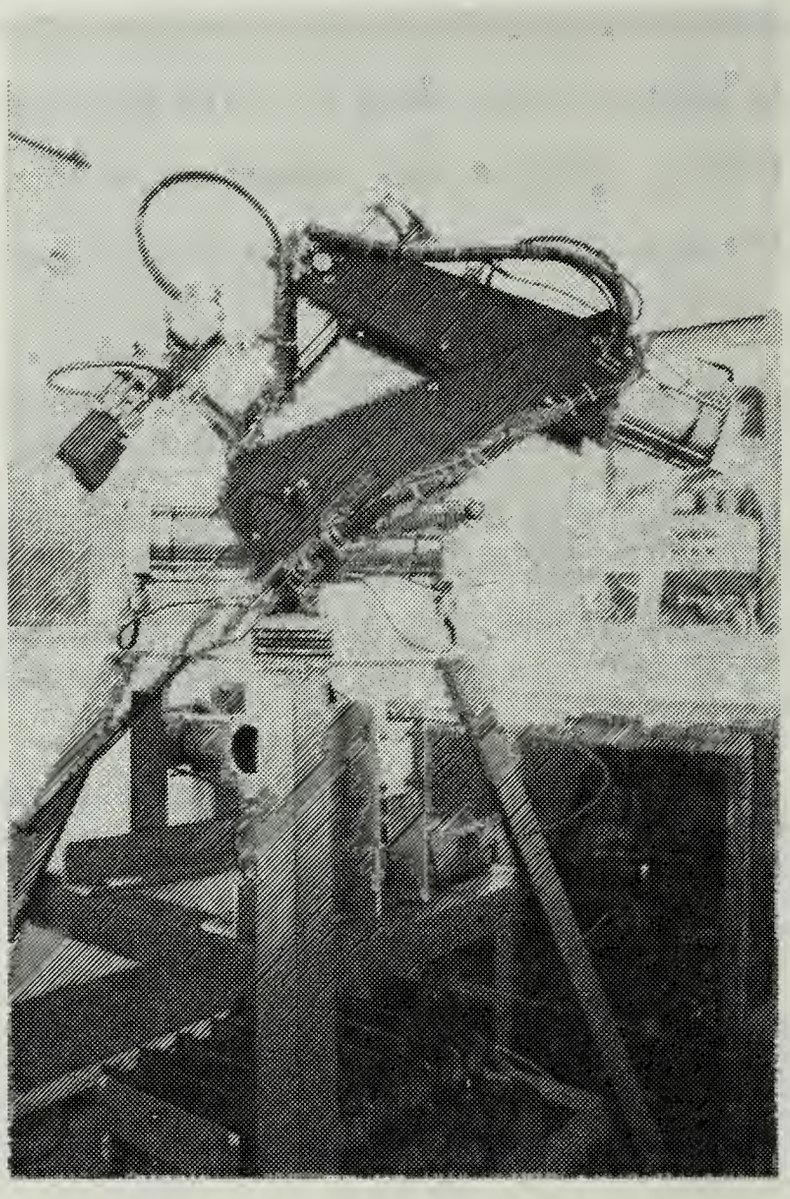
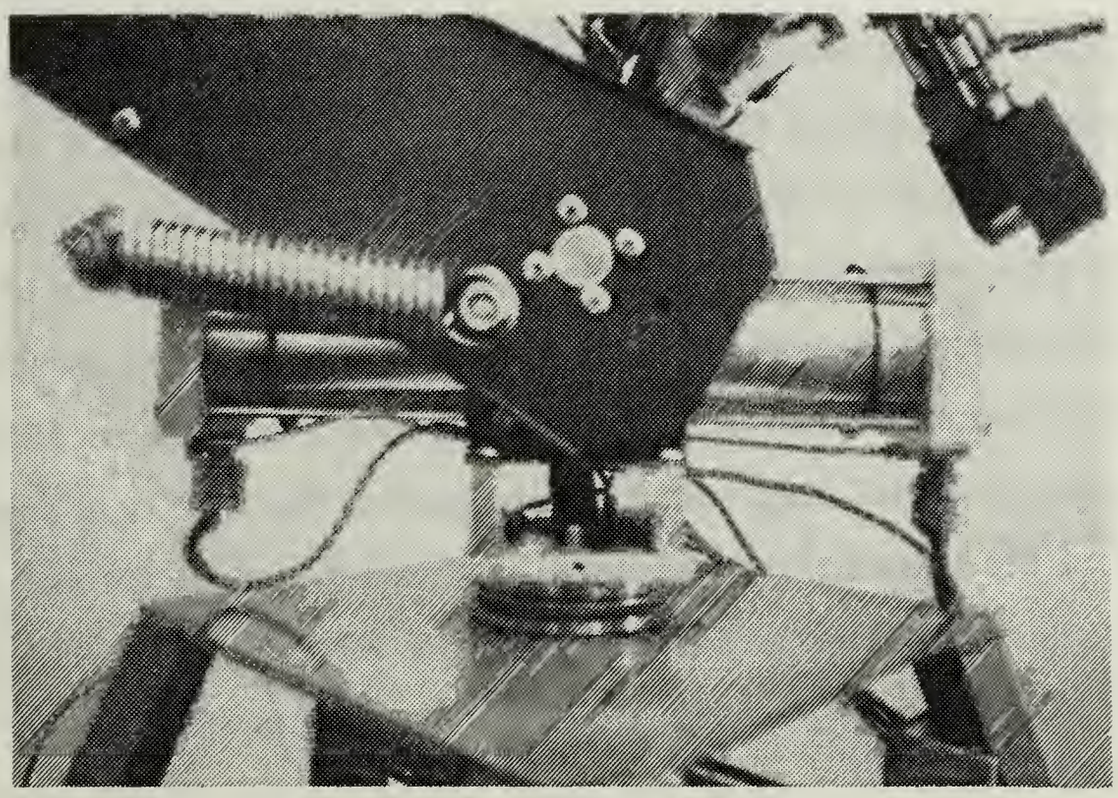
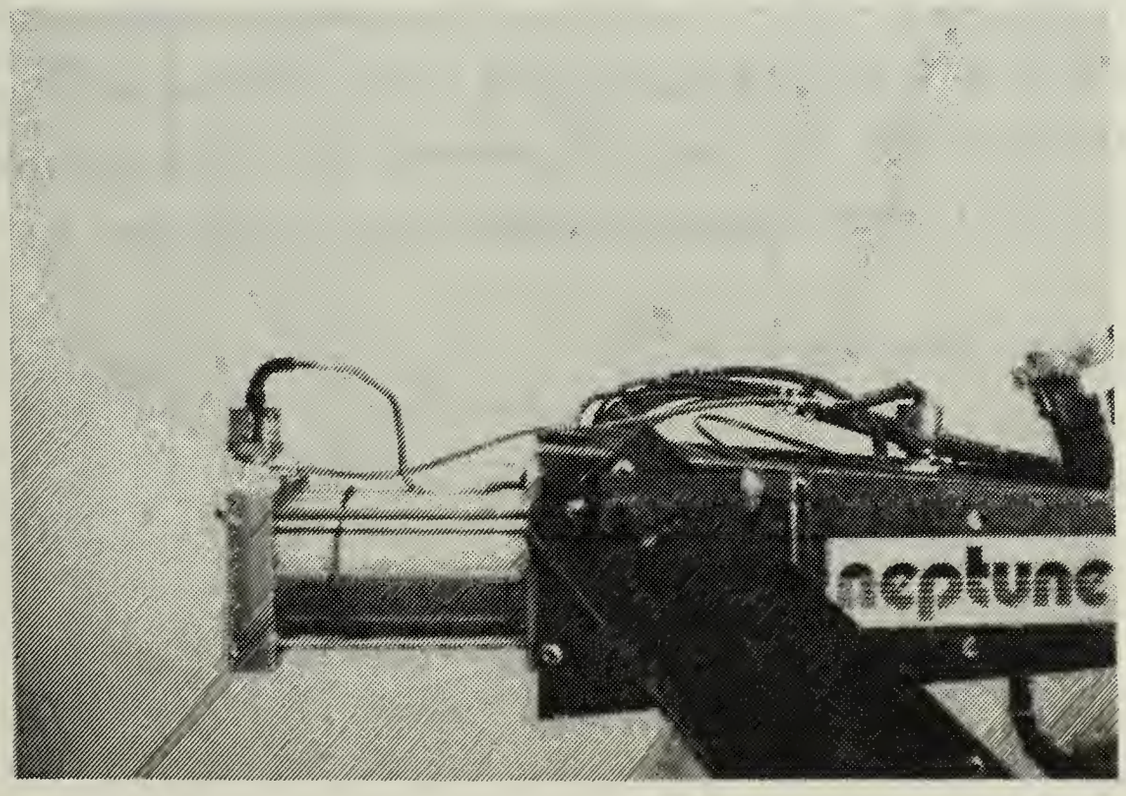


Figure 15. **NEPTUNE II Robot**



Joint 1



Joint 2

Figure 16. **Robot Pressure Transducers**

A 3,000-pound nitrogen bottle with a 200 psi regulator was used for pressure transducer calibration. This was connected to a test rig which consisted of a varlinear pressure adjustment and a 200-pound pressure gage which had been calibrated and was incremented in 0.2-pound increments (Figure 17). The varlinear allowed the pressure at the transducer to be varied in fine increments from zero to 120 psi, which was the stated range of the transducers. The electrical connections from the transducers had been hard-wired to the ECP and then to a digital multimeter (Fluke 8024 B); the meter 20-volt scale was used to read the voltage output of the transducers (the voltage was read to the second decimal place). Each Omega Series PX302 pressure transducer had two complete sets of data from 0 to 120 psi; the averages of these are provided in Table 1. The pressure transducers were very linear from 0 to 115 psi, with each 5 psi producing a voltage of 0.5 volts (Figure 18). This range of pressures was well acceptable for the NEPTUNE II, which operates between 0 and 118 psi [Ref. 12].

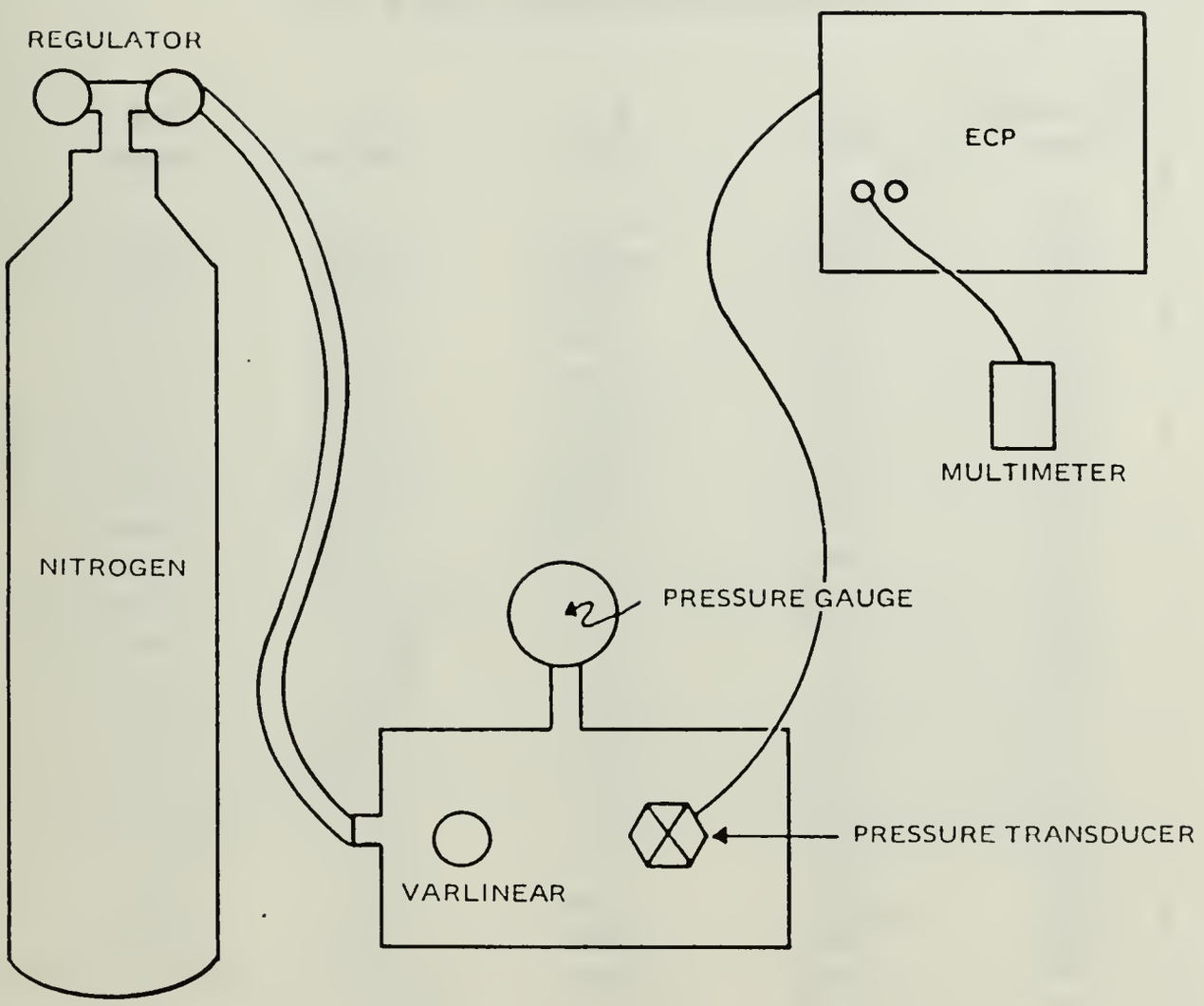


Figure 17. **Pressure Transducer Calibration Set-Up**

TABLE 1

PRESSURE TRANSDUCER DATA

Pressure (psi)	P1 volts	P2 volts	P3 volts	P4 volts
000	-0.010	0.000	0.015	0.000
005	0.500	0.510	0.520	0.500
010	0.995	1.005	1.025	1.005
015	1.500	1.500	1.530	1.500
020	1.985	1.995	2.020	2.000
025	2.500	2.500	2.530	2.500
030	3.000	3.000	3.030	3.000
035	3.490	3.500	3.530	3.490
040	3.990	3.995	4.030	3.995
045	4.500	4.500	4.530	4.500
050	4.995	4.995	5.025	4.990
055	5.490	5.500	5.520	5.490
060	5.995	5.995	6.020	5.990
065	6.490	6.500	6.520	6.490
070	6.990	7.000	7.020	6.990
075	7.490	7.500	7.510	7.500
080	7.990	8.005	8.010	8.000
085	8.490	8.510	8.510	8.500
090	8.990	9.005	9.000	9.000
095	9.480	9.500	9.500	9.500
100	9.990	10.005	10.000	10.000
105	10.490	10.510	10.500	10.500
110	10.990	11.010	11.000	11.000
115	11.485	11.490	11.490	11.500
120	11.580	11.490	11.990	12.000

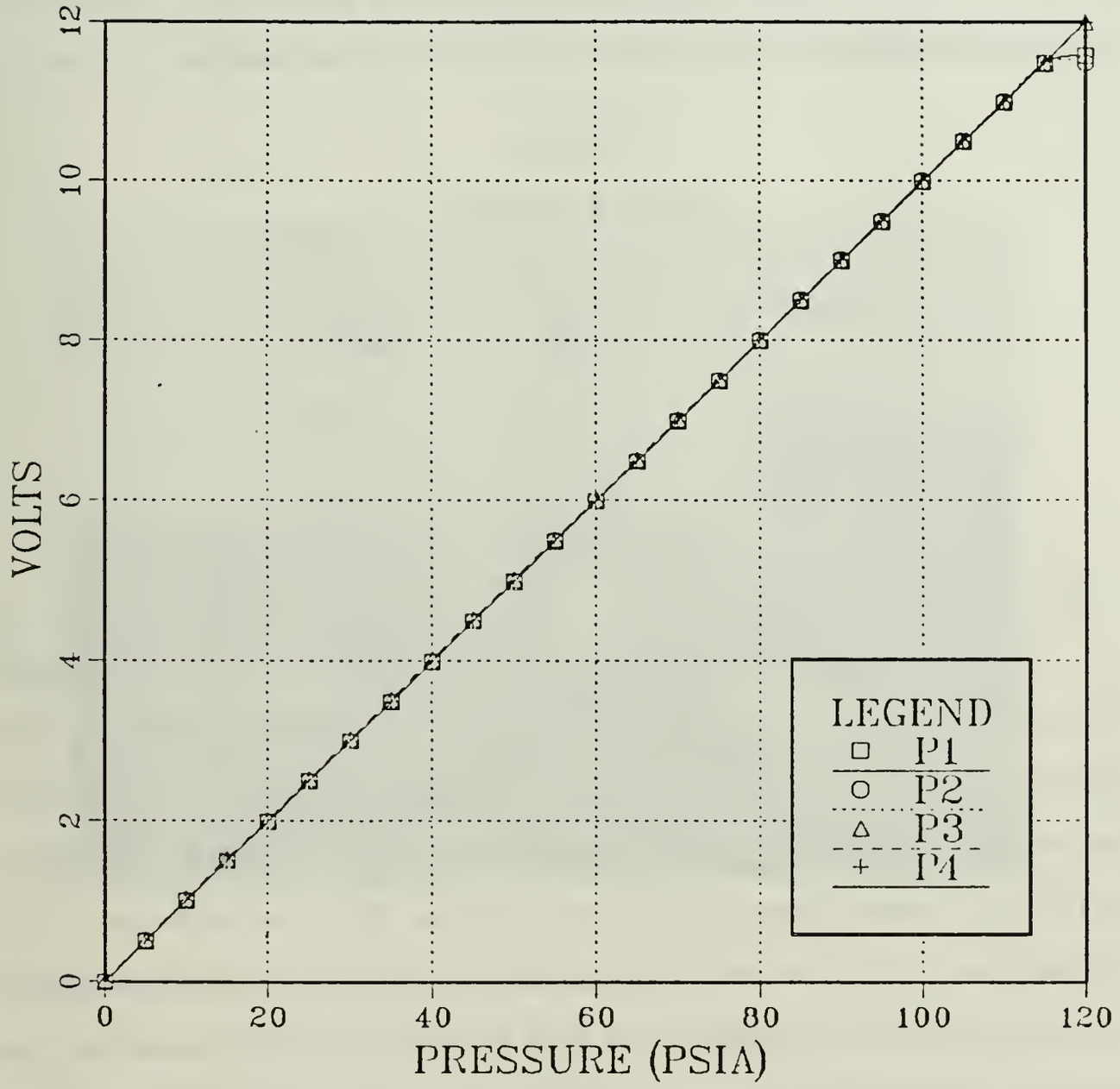


Figure 18. **Pressure Transducer Data**

C. MEASUREMENTS

1. Link Lengths

With the necessity to disassemble the NEPTUNE II for repairs, the link length measurements were physically measured. These measurements are provided in Table 2 to the nearest 1/8 inch.

TABLE 2

LINK LENGTHS

Parameter	Length inch	Length meter
LINKL1	10.250	0.2604
LINKL2	16.375	0.4159
LINKL3	13.750	0.3498

2. Link Masses

Also during the disassembly of the robot, the link masses were measured using a scale which was in half-pound increments. The design of the robot allowed for a good approximation of the two masses per link as used in the simulation (Figures 19 and 20). This was a crude measurement, but it was felt that this was sufficient to be used for the simulation and that the time necessary to completely disassemble the robot in order to be able to obtain closer measurements was not justified. The masses which were measured are provided in Table 3.

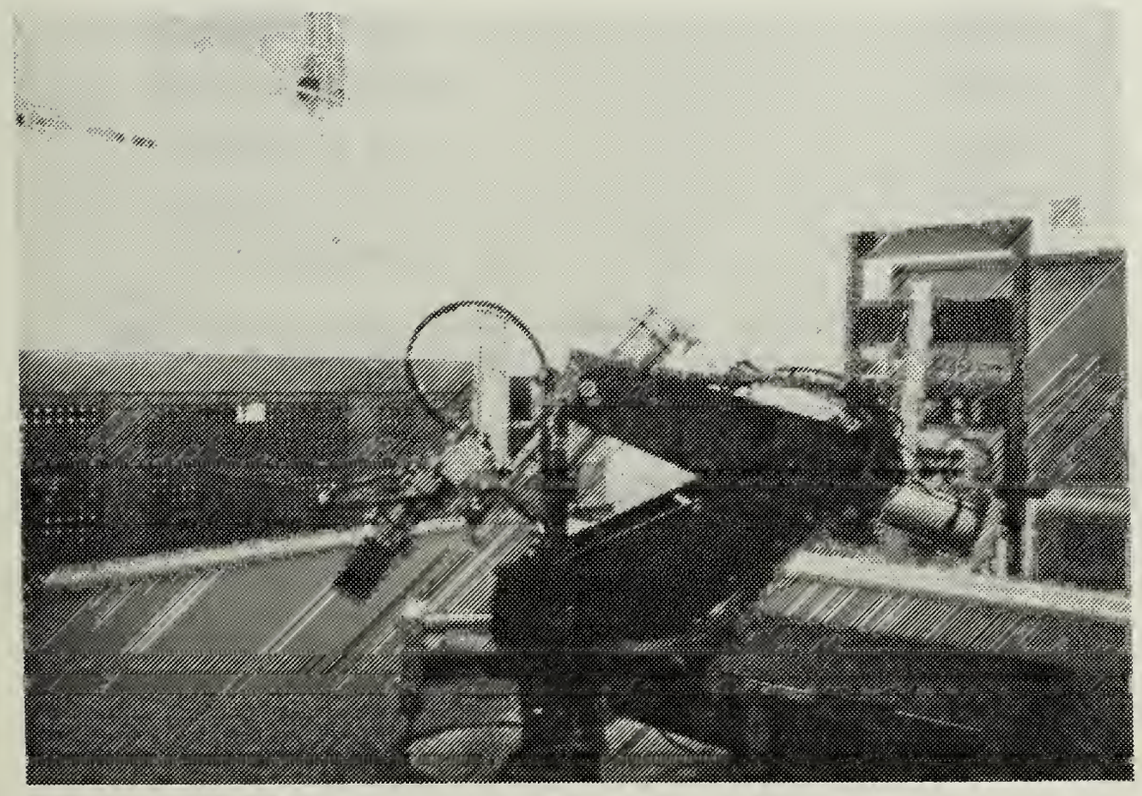


Figure 19. **Robot Link Masses**

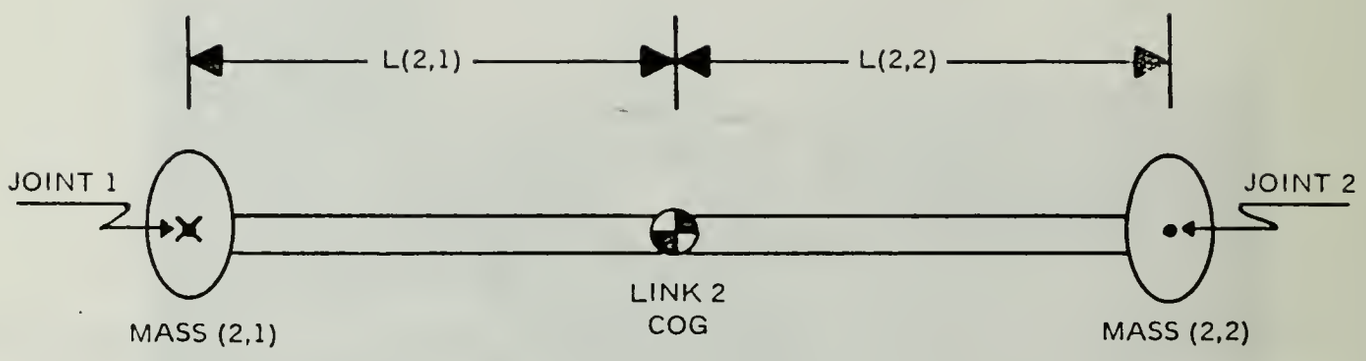


Figure 20. **Simulation Link Masses**

TABLE 3

LINK MASSES

Parameter	Pounds	Kilograms
MASS(1,1) est.	5	2.273
MASS(1,2) est.	15	6.818
M1 (measured)	20	9.091
MASS(2,1) est.	1	0.455
MASS(2,2) est.	1	0.455
M2 (measured)	2	0.910
MASS(3,1) est.	13	5.909
MASS(3,2) est.	13	5.909
M3 (measured)	26	11.818

3. Length from COG of Link to Center of Mass

The point mass centers were assumed to be at the joints (Figure 20). A simple calculation of the moments around the center of gravity was performed; this provided the data in Table 4.

TABLE 4

LENGTH FROM LINK CENTER OF GRAVITY

TO MASS CENTER OF GRAVITY

Parameter	Length inch	Length meter
L(1,1)	7.6875	0.1953
L(1,2)	2.5625	0.0651
L(2,1)	8.1875	0.2080
L(2,2)	8.1875	0.2080
L(3,1)	6.8750	0.1746
L(3,2)	6.8750	0.1746

D. DATA COLLECTION

Position data and pressure data were taken for several movements of links 2 and 3 of the robot from the electronic control panel and recorded on a four-channel strip chart recorder. An example of the recorded data is provided (Figure 21).

E. DATA REDUCTION

The data taken for the various runs was reduced using the following equation to obtain joint torques.

$$\text{TORQUE} = \Delta \text{ PRESSURE} * \text{AREA(piston)} * \text{RADIUS(lever arm)} \quad (51)$$

where AREA(piston) = 0.00312 square meters and RADIUS(lever arm) = 0.0597/2 meters.

F. VALIDATION

The delta pressure data versus time was entered into the robot validation program as a time function and the torque was calculated as a function of time from this data. The simulation position results agreed with the actual robot position in the general trend of motion (Figure 22). It was felt that the inaccuracies in the results were due to both the simulation round-off errors and the crude methods used in the collection of the robot link masses and link lengths, particularly the lengths from the center of gravity of the link to the center of gravity of the point masses. The scalloped shape of the simulated result was attributed to the normalization of the directional cosines.

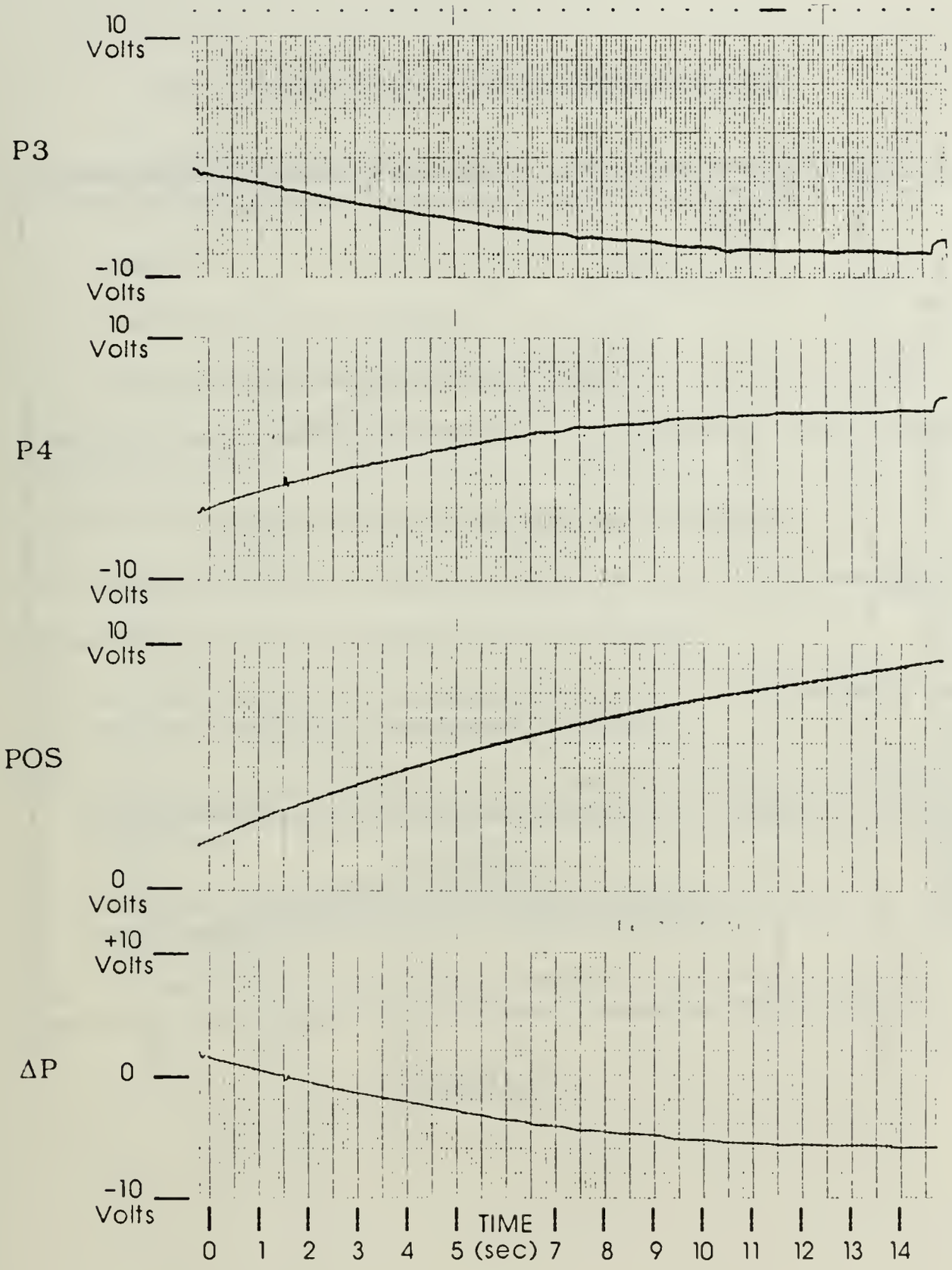


Figure 21. **Pressure Recordings**

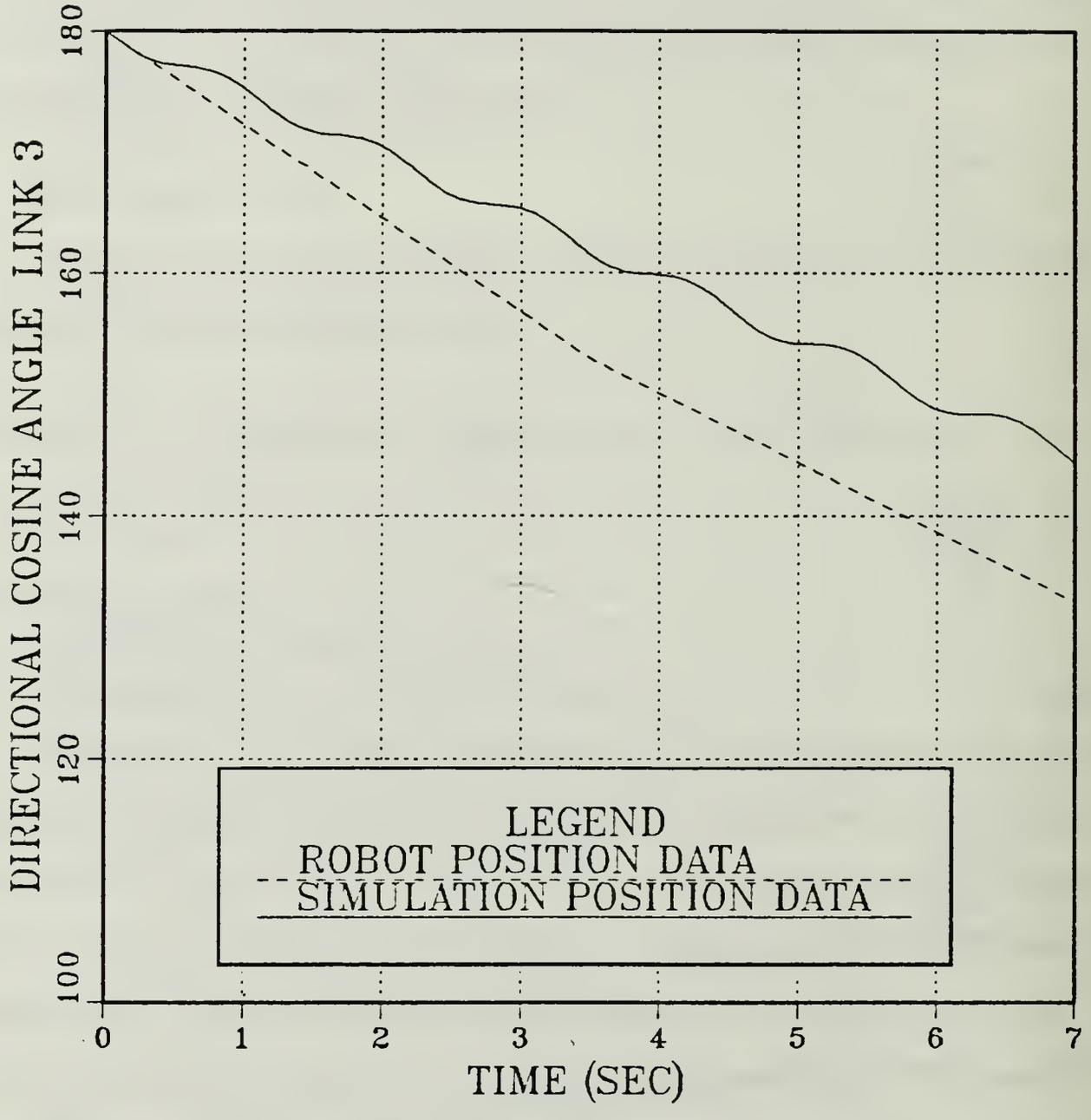


Figure 22. **Robot/Simulation Motion**

VI. RESULTS AND RECOMMENDATIONS

1. A nonsingular dynamic simulation of a rigid revolute three-link manipulator has been further developed. It includes:
   - Gravity,
   - Three-Dimensional Motion,
   - Comparatively straightforward dynamics.
2. Accuracy and nonsingularity has been validated with a double pendulum.
3. Three-dimensional nonsingularity has been validated.
4. Procedures for actual robot data collection have been developed.
5. The following recommendations are provided:
   - Simulate actual three-dimensional robot performance and compare predicted to measured variables.
   - Enhance the computer program code to enable it to be more interactive. That is, let the user specify the constraints during execution.
   - Use this simulation to develop advanced controllers for rigid robot linkages.

APPENDIX A

DIRECT DYNAMIC SIMULATION PROGRAM

```
TITLE DIRECT DYNAMICS PROBLEM

**********************************************************************************
*  THIS IS A PROGRAM THAT WILL SOLVE THE DIRECT DYNAMICS PROBLEM FOR A  *
*  3 LINK RIGID BODY MANIPULATOR WITHOUT THE USE OF ANY TRANSFORMATION  *
*  MATRICES.  THE ORIGINAL PROGRAM WAS WRITTEN BY LT ATINOK AND HAS     *
*  HAS BEEN GREATLY MODIFIED TO INCLUDE 3 D MOTION AND GRAVITATIONAL    *
*  EFFECTS BY      LT  R. M.  VERBOS   USN       SEPTEMBER 1988         *
**********************************************************************************

TERMINAL
METHOD RKSFX
PRINT .10,DRCANX(1),DRCANY(1),DRCANZ(1),...
DRCANX(2),DRCANY(2),DRCANZ(2),DRCANX(3),DRCANY(3),DRCANZ(3),...
DRCSX(1),DRCSY(1),DRCSZ(1),DRCSX(2),DRCSY(2),DRCSZ(2),DRCSX(3),...
DRCSY(3),DRCSZ(3),...
JX0,JY0,JZ0,JX1,JY1,JZ1,JX2,JY2,JZ2,...
LCOGX(1),LCOGY(1),LCOGZ(1),LCOGX(2),LCOGY(2),LCOGZ(2),...
LCOGX(3),LCOGY(3),LCOGZ(3),...
T0X,T0Y,T0Z,T1X,T1Y,T1Z,T2X,T2Y,T2Z,...
WDX(1),WDY(1),WDZ(1),WDX(2),WDY(2),WDZ(2),WDX(3),WDY(3),WDZ(3),...
AX1,AY1,AZ1,AX2,AY2,AZ2,AX3,AY3,AZ3,...
W1X,W1Y,W1Z,W2X,W2Y,W2Z,W3X,W3Y,W3Z,...
FX0,FY0,FZ0,FX1,FY1,FZ1,FX2,FY2,FZ2,...
IXXT(1),IYYT(1),IZZT(1),IXXT(2),IYYT(2),IZZT(2),IXXT(3),IYYT(3),...
IZZT(3),IXYT(1),IYZT(1),IXZT(1),IXYT(2),IYZT(2),IXZT(2),IXYT(3),...
IYZT(3),IXZT(3),...
LNCOG1,LNCOG2,LNCOG3,THZANG,COSTHZ,SINTHZ,TIPX,TIPY,TIPZ,...
ANG12X,ANG12Y,ANG12Z,ANG23X,ANG23Y,ANG23Z,...
T1CH,T2CH,IER
CONTROL FINTIM =1.5, DELT = .0005, DELMAX = .1, DELPRT = .10
D     DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D     DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D     DIMENSION IMAT(3,3,3),NIMAT(3,3),TMAT(3,3),TMATTR(3,3),MATTMP(3,3)
D     DIMENSION LIMAT(3,3,3)
FIXED   IER,I,J,M,K,P,N,IA,IDGT,A,I1
ARRAY MATB(27),ABMATB(27),LCOGX(3),LCOGY(3),LCOGZ(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY VECTA(3),VECTB(3)
ARRAY WDX(3),WDY(3),WDZ(3),W1(3),W2(3),W3(3)
ARRAY RBG1(3),RAG1(3),RBG2(3),RAG2(3),RBG3(3)
ARRAY WKAREA(850)
ARRAY IXXT(3),IYYT(3),IZZT(3),IXYT(3),IXZT(3),IYZT(3)
```

```
ARRAY DRCANX(3),DRCANY(3),DRCANZ(3)
ARRAY DRCSX(3),DRCSY(3),DRCSZ(3)

INITIAL

******   INPUT   ******
*       INPUT PARAMETER CONSTANTS
55          A = 15.0D0
            P = 0.0D0
            W =  2 * PI
            IDGT = 6
           G = 9.81D0
           N = 27
           M = 1
           IA = 27
           IER = 0
           LEVELQ = 0
           LEVLDQ = 0

*       INPUT JOINT LOCATIONS IN METERS
           JX0 = 0.0D0
           JY0 = 0.0D0
           JZ0 = 0.0D0
           JX1 = 0.0D0
           JY1 = 0.0D0
           JZ1 = 0.2D0
           JX2 = 0.0D0
           JY2 = 0.416D0
           JZ2 = 0.2D0

*       INPUT TORQUE CONSTANTS
           T0X    = 0.0D0
           T0Y    = 0.0D0
           T0Z    = 0.0D0
           T1X    = 0.0D0
           T1Y    = 0.0D0
           T1Z    = 0.0D0
           T2X    = 0.0D0
           T2Y    = 0.0D0
           T2Z    = 0.0D0
           TG1    = 0.0D0
           TG2    = 0.0D0
           T1FNC  = 0.0D0
           T2FNC  = 0.0D0

*       INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS
*       FOR EACH LINK ENDS
           L(1,1) = 0.05D0
           L(1,2) = 0.15D0
           L(2,1) = 0.20D0
           L(2,2) = 0.216D0
           L(3,1) = 0.225D0
           L(3,2) = 0.226D0

* INPUT THE LINK LENGTHS OF THE ROBOT
           LINKL1 = 0.20D0
```

```
          LINKL2 = 0.416D0
          LINKL3 = 0.451D0

*       INPUT MASS AT LINK ENDS IN KILOGRAMS
110       MASS(1,1) = 11.0D0
          MASS(1,2) = 33.0D0
          MASS(2,1) = 2.2D0
          MASS(2,2) = 2.2D0
          MASS(3,1) = 28.6D0
          MASS(3,2) = 28.6D0

*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
120         LCOGX(1) =  0.0D0
            X1       =  LCOGX(1)
            LCOGY(1) =  0.0D0
            Y1       =  LCOGY(1)
            LCOGZ(1) =  0.10D0
            Z1       =  LCOGZ(1)
            LCOGX(2) =  0.0D0
            X2       =  LCOGX(2)
            LCOGY(2) =  0.208D0
            Y2       =  LCOGY(2)
            LCOGZ(2) =  0.20D0
            Z2       =  LCOGZ(2)
            LCOGX(3) =  0.0D0
            X3       =  LCOGX(3)
            LCOGY(3) =  0.6415D0
            Y3       =  LCOGY(3)
            LCOGZ(3) =  0.2D0
            Z3       =  LCOGZ(3)

*       INPUT MASS OF EACH LINK IN KG
            M1 = 44.0D0
            M2 = 4.4D0
            M3 = 57.2D0

*       INPUT ACCELERATIONS OF JOINT ZERO
            AX0 = 0.0D0
            AY0 = 0.0D0
            AZ0 = 0.0D0

*       INPUT THE INITIAL DIRECTION COSINES
            DRCSX(1) = 0.0D0
            DRCSY(1) = 0.0D0
            DRCSZ(1) = 1.0D0
            DRCSX(2) = 0.0D0
            DRCSY(2) = 1.0D0
            DRCSZ(2) = 0.0D0
            DRCSX(3) = 0.0D0
            DRCSY(3) = 1.0D0
            DRCSZ(3) = 0.0D0

*       INPUT  THE INITIAL DIRECTION COSINE ANGLES
            DRCANX(1) = 90.0D0
            DRCANY(1) = 90.0D0
            DRCANZ(1) =  0.0D0
```

```
           DRCANX(2) = 90.0D0
           DRCANY(2) =  0.0D0
           DRCANZ(2) = 90.0D0
           DRCANX(3) = 90.0D0
           DRCANY(3) =  0.0D0
           DRCANZ(3) = 90.0D0

******   INITIALIZE   ******
*       OMEGA AND OMEGA DOT
160        DO 170 I = 1,3
              W1(I)     = 0.0D0
              W2(I)     = 0.0D0
              W3(I)     = 0.0D0
              WDX(I)    = 0.0D0
              WDY(I)    = 0.0D0
              WDZ(I)    = 0.0D0
170        CONTINUE

           THZ = 0.0D0

*      INITIALIZE MATRIX A AND B TO ZERO
           DO 180 I = 1,27
              DO 175 J = 1,27
                 MATA(I,J) = 0.0D0
175           CONTINUE
               MATB(I) = 0.0D0
180        CONTINUE

******    CALCULATIONS    ******
*       WEIGHTS (NEWTONS)
185        WG1 = M1*G
           WG2 = M2*G
           WG3 = M3*G

*       COMPUTE THE LENGTH FROM THE INBOARD JOINT TO COG
           LNCOG1 = DSQRT ( LCOGX(1)*LCOGX(1) + LCOGY(1)*LCOGY(1) +...
                            LCOGZ(1)*LCOGZ(1) )
           LX2 = LCOGX(2) - JX1
           LY2 = LCOGY(2) - JY1
           LZ2 = LCOGZ(2) - JZ1
           LNCOG2 = DSQRT ( LX2*LX2 + LY2*LY2 + LZ2*LZ2 )
           LX3 = LCOGX(3) - JX2
           LY3 = LCOGY(3) - JY2
           LZ3 = LCOGZ(3) - JZ2
           LNCOG3 = DSQRT ( LX3*LX3 + LY3*LY3 + LZ3*LZ3 )

*       COMPUTE INITIAL INERTIAS BASED ON POINT MASSES
*                IN GLOBAL COORDINATES
190        DO 225 I = 1,3
              RX(I,1) = -L(I,1) * DRCSX(I)
              RX(I,2) =  L(I,2) * DRCSX(I)
              RY(I,1) = -L(I,1) * DRCSY(I)
              RY(I,2) =  L(I,2) * DRCSY(I)
              RZ(I,1) = -L(I,1) * DRCSZ(I)
              RZ(I,2) =  L(I,2) * DRCSZ(I)
200    IXX(I,1) = MASS(I,1) * ((RY(I,1) * RY(I,1)) + (RZ(I,1) * RZ(I,1)))
```

```
      IXX(I,2) = MASS(I,2) * ((RY(I,2) * RY(I,2)) + (RZ(I,2) * RZ(I,2)))
      IXXT(I)  = IXX(I,1) + IXX(I,2)
      IYY(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RZ(I,1) * RZ(I,1)))
      IYY(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RZ(I,2) * RZ(I,2)))
      IYYT(I)  = IYY(I,1) + IYY(I,2)
      IZZ(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RY(I,1) * RY(I,1)))
      IZZ(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RY(I,2) * RY(I,2)))
      IZZT(I)  = IZZ(I,1) + IZZ(I,2)
205   IXY(I,1) = MASS(I,1) * RX(I,1) * RY(I,1)
      IXY(I,2) = MASS(I,2) * RX(I,2) * RY(I,2)
      IXYT(I)  = IXY(I,1) + IXY(I,2)
      IXZ(I,1) = MASS(I,1) * RZ(I,1) * RX(I,1)
      IXZ(I,2) = MASS(I,2) * RZ(I,2) * RX(I,2)
      IXZT(I)  = IXZ(I,1) + IXZ(I,2)
      IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
      IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
      IYZT(I)  = IYZ(I,1) + IYZ(I,2)
      IF (IXXT(I) .LE. .02) THEN
      IXXT(I)  = .02
      ELSE
      IXXT(I) = IXXT(I)
      END IF
      IF (IYYT(I) .LE. .02) THEN
      IYYT(I)  = .02
      ELSE
      IYYT(I)  = IYYT(I)
      END IF
      IF (IZZT(I) .LE. .02) THEN
      IZZT(I)  = .02
      ELSE
      IZZT(I)  = IZZT(I)
      END IF
           IMAT(I,1,1) =  IXXT(I)
           IMAT(I,1,2) =  IXYT(I)
           IMAT(I,1,3) =  IXZT(I)
           IMAT(I,2,1) = -IXYT(I)
           IMAT(I,2,2) =  IYYT(I)
           IMAT(I,2,3) =  IYZT(I)
           IMAT(I,3,1) = -IXZT(I)
           IMAT(I,3,2) = -IYZT(I)
           IMAT(I,3,3) =  IZZT(I)
225        CONTINUE

*      DUE TO LINK 1 CONSTRAINTS LINK 1 INERTIAS ARE CONSTANT
         IXXT(1) = IMAT(1,1,1)
         IXYT(1) = IMAT(1,1,2)
         IXZT(1) = IMAT(1,1,3)
         IYYT(1) = IMAT(1,2,2)
         IYZT(1) = IMAT(1,2,3)
         IZZT(1) = IMAT(1,3,3)

*    TRANSFORM THE INITIAL INERTIAS TO LOCAL COORDINATED
       DO 9 I = 2, 3
         TMAT(2,1) = -DCOS ( THZ )
         TMAT(2,2) = -DSIN ( THZ )
         TMAT(2,3) = 0.0D0
```

```
            TMAT(3,1) = DRCSX(I)
            TMAT(3,2) = DRCSY(I)
            TMAT(3,3) = DRCSZ(I)

            VECTA(1)  = TMAT(2,1)
            VECTA(2)  = TMAT(2,2)
            VECTA(3)  = TMAT(2,3)
            VECTB(1)  = TMAT(3,1)
            VECTB(2)  = TMAT(3,2)
            VECTB(3)  = TMAT(3,3)
            CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
            TMAT(1,1) = MI1
            TMAT(1,2) = MJ1
            TMAT(1,3) = MK1
            TMATTR(1,1) = TMAT(1,1)
            TMATTR(1,2) = TMAT(2,1)
            TMATTR(1,3) = TMAT(3,1)
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)

            DO 5 I1 = 1,3
            DO 5 J  = 1,3
               TEMP = 0.0D0
              DO 1  K  = 1,3
                 TEMP = TMAT(I1,K) * IMAT(I,K,J) + TEMP
1             CONTINUE
               MATTMP(I1,J) = TEMP
5           CONTINUE

            DO 8   I1 = 1,3
            DO 8   J  = 1,3
               TEMP = 0.0D0
              DO 7   K  = 1,3
                 TEMP = MATTMP(I1,K) * TMATTR(K,J) + TEMP
7             CONTINUE
             LIMAT(I.I1,J) = TEMP
8           CONTINUE
9         CONTINUE

DERIVATIVE
NOSORT
230         CALL ERRSET (208,256,-1,1,1)
            CALL UERSET(LEVELQ,LEVLDQ)
*     INITIALIZE MATRIX A AND B TO ZERO
            DO 240 I = 1,27
               DO 235 J = 1,27
                  MATA(I,J) = 0.0D0
235           CONTINUE
               MATB(I) = 0.0D0
240         CONTINUE

*       INPUT JOINT EQUATIONS
```

```
*       JOINT ZERO  EQUATIONS
*       W1 X (W1 X RB/G1)
              RBG1(1) = JX0 - LCOGX(1)
              RBG1(2) = JY0 - LCOGY(1)
              RBG1(3) = JZ0 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD(VECTA0,RBG1,MIC0,MJC0,MKC0)
              VECTB0(1) = MIC0
              VECTB0(2) = MJC0
              VECTB0(3) = MKC0
           CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)

*       INPUT JOINT EQUATIONS
*       JOINT ZERO  EQUATIONS
*        W1 X (W1 X RB/G1)
              RBG1(1)  = JX0 - LCOGX(1)
              RBG1(2)  = JY0 - LCOGY(1)
              RBG1(3)  = JZ0 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD(VECTA,RBG1,MIC0,MJC0,MKC0)
              VECTB(1) = MIC0
              VECTB(2) = MJC0
              VECTB(3) = MKC0
           CALL CPROD(VECTA,VECTB,MIC0,MJC0,MKC0)
*       W1 X (W1 X RA/G1)
              RAG1(1) = JX1 - LCOGX(1)
              RAG1(2) = JY1 - LCOGY(1)
              RAG1(3) = JZ1 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD (VECTA,RAG1,MIC1,MJC1,MKC1)
              VECTB(1) = MIC1
              VECTB(2) = MJC1
              VECTB(3) = MKC1
           CALL CPROD (VECTA,VECTB,MIC1,MJC1,MKC1)
*       W2 X (W2 X RB/G2)
              RBG2(1) = JX1 - LCOGX(2)
              RBG2(2) = JY1 - LCOGY(2)
              RBG2(3) = JZ1 - LCOGZ(2)
              VECTA(1) = W2(1)
              VECTA(2) = W2(2)
              VECTA(3) = W2(3)
           CALL CPROD (VECTA,RBG2,MIC2,MJC2,MKC2)
              VECTB(1) = MIC2
              VECTB(2) = MJC2
              VECTB(3) = MKC2
           CALL CPROD (VECTA,VECTB,MIC2,MJC2,MKC2)
*       W2 X (W2 X RA/G2)
              RAG2(1) = JX2 - LCOGX(2)
              RAG2(2) = JY2 - LCOGY(2)
```

```
            RAG2(3) = JZ2 - LCOGZ(2)
            VECTA(1) = W2(1)
            VECTA(2) = W2(2)
            VECTA(3) = W2(3)
         CALL CPROD (VECTA,RAG2,MIC3,MJC3,MKC3)
            VECTB(1) = MIC3
            VECTB(2) = MJC3
            VECTB(3) = MKC3
         CALL CPROD(VECTA,VECTB,MIC3,MJC3,MKC3)
*     W3 X (W3 X RB/G3)
            RBG3(1) = JX2 - LCOGX(3)
            RBG3(2) = JY2 - LCOGY(3)
            RBG3(3) = JZ2 - LCOGZ(3)
            VECTA(1) = W3(1)
            VECTA(2) = W3(2)
            VECTA(3) = W3(3)
         CALL CPROD (VECTA,RBG3,MIC4,MJC4,MKC4)
            VECTB(1) = MIC4
            VECTB(2) = MJC4
            VECTB(3) = MKC4
         CALL CPROD (VECTA,VECTB,MIC4,MJC4,MKC4)

*     INERTIA TRANSFORMATION
        DO 390 I = 2, 3
          TMAT(2,1) = -DCOS ( THZ )
          TMAT(2,2) = -DSIN ( THZ )
          TMAT(2,3) = 0.0D0
          TMAT(3,1) = DRCSX(I)
          TMAT(3,2) = DRCSY(I)
          TMAT(3,3) = DRCSZ(I)

          VECTA(1)  = TMAT(2,1)
          VECTA(2)  = TMAT(2,2)
          VECTA(3)  = TMAT(2,3)
          VECTB(1)  = TMAT(3,1)
          VECTB(2)  = TMAT(3,2)
          VECTB(3)  = TMAT(3,3)
          CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
          TMAT(1,1) = MI1
          TMAT(1,2) = MJ1
          TMAT(1,3) = MK1
          TMATTR(1,1) = TMAT(1,1)
          TMATTR(1,2) = TMAT(2,1)
          TMATTR(1,3) = TMAT(3,1)
          TMATTR(2,1) = TMAT(1,2)
          TMATTR(2,2) = TMAT(2,2)
          TMATTR(2,3) = TMAT(3,2)
          TMATTR(3,1) = TMAT(1,3)
          TMATTR(3,2) = TMAT(2,3)
          TMATTR(3,3) = TMAT(3,3)

          DO 375 I1 = 1,3
          DO 375 J  = 1,3
             TEMP = 0.0D0
            DO 370  K  = 1,3
```

```
                TEMP = TMATTR(I1,K) * LIMAT(I,K,J) + TEMP
370           CONTINUE
               MATTMP(I1,J) = TEMP
375         CONTINUE

            DO 380 I1 = 1,3
            DO 380 J  = 1,3
               TEMP = 0.0D0
              DO 378 K  = 1,3
                 TEMP = MATTMP(I1,K) * TMAT(K,J) + TEMP
378           CONTINUE
             NIMAT(I1,J) = TEMP
380         CONTINUE

             IXXT(I) = NIMAT(1,1)
             IXYT(I) = NIMAT(1,2)
             IXZT(I) = NIMAT(1,3)
             IYYT(I) = NIMAT(2,2)
             IYZT(I) = NIMAT(2,3)
             IZZT(I) = NIMAT(3,3)

390     CONTINUE

******  ENTER CONSTANTS INTO MATRIX A AND B  ******
***     LINK ONE
*       SUM OF FORCES
*       X DIRECTION
395       MATA(1,1)  =  1.0D0
          MATA(1,4)  =  M1
          MATA(1,10) = -1.0D0
          MATB(1)    =  0.0D0
*       Y DIRECTION
          MATA(2,2)  =  1.0D0
          MATA(2,5)  =  M1
          MATA(2,11) = -1.0D0
          MATB(2)    =  0.0D0
*       Z DIRECTION
          MATA(3,3)  =  1.0D0
          MATA(3,6)  =  M1
          MATA(3,12) = -1.0D0
          MATB(3)    = -WG1
*       EQUATIONS AT JOINT ZERO
*       X DIRECTION
          MATA(4,4) =  1.0D0
          MATA(4,8) =  RBG1(3)
          MATA(4,9) = -RBG1(2)
          MATB(4)   =  AX0 - MIC0
*       Y DIRECTION
          MATA(5,5) =  1.0D0
          MATA(5,7) = -RBG1(3)
          MATA(5,9) =  RBG1(1)
          MATB(5)   =  AY0 - MJC0
*       Z DIRECTION
          MATA(6,6) =  1.0D0
          MATA(6,7) =  RBG1(2)
          MATA(6,8) = -RBG1(1)
```

```
        MATB(6)   =  AZ0 - MKC0
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
        MATA(7,2)  =  RBG1(3)
        MATA(7,3)  = -RBG1(2)
        MATA(7,7)  = -IXXT(1)
        MATA(7,8)  =  IXYT(1)
        MATA(7,9)  =  IXZT(1)
        MATA(7,11) = -RAG1(3)
        MATA(7,12) =  RAG1(2)
        MATB(7)    =  T1X - T0X
*       Y DIRECTION
        MATA(8,1)  = -RBG1(3)
        MATA(8,3)  =  RBG1(1)
        MATA(8,7)  =  IXYT(1)
        MATA(8,8)  = -IYYT(1)
        MATA(8,9)  =  IYZT(1)
        MATA(8,10) =  RAG1(3)
        MATA(8,12) = -RAG1(1)
        MATB(8)    =  T1Y - T0Y
*       Z DIRECTION
        MATA(9,1)  =  RBG1(2)
        MATA(9,2)  = -RBG1(1)
        MATA(9,7)  =  IXZT(1)
        MATA(9,8)  =  IYZT(1)
        MATA(9,9)  = -IZZT(1)
        MATA(9,10) = -RAG1(2)
        MATA(9,11) =  RAG1(1)
        MATB(9)    =  T1Z - T0Z
**      LINK TWO
*       SUM OF FORCES
*       X DIRECTION
460     MATA(10,10) =  1.0D0
        MATA(10,13) =  M2
        MATA(10,19) = -1.0D0
        MATB(10)    =  0.0D0
*       Y DIRECTION
        MATA(11,11) =  1.0D0
        MATA(11,14) =  M2
        MATA(11,20) = -1.0D0
        MATB(11)    =  0.0D0
*       Z DIRECTION
        MATA(12,12) =  1.0D0
        MATA(12,15) =  M2
        MATA(12,21) = -1.0D0
        MATB(12)    = -WG2
*       EQUATIONS AT JOINT ONE
*       X DIRECTION
        MATA(13,4)  = -1.0D0
        MATA(13,8)  = -RAG1(3)
        MATA(13,9)  =  RAG1(2)
        MATA(13,13) =  1.0D0
        MATA(13,17) =  RBG2(3)
        MATA(13,18) = -RBG2(2)
        MATB(13)    =  MIC1 - MIC2
*       Y DIRECTION
```

```
          MATA(14,5)  = -1.0D0
          MATA(14,7)  =  RAG1(3)
          MATA(14,9)  = -RAG1(1)
          MATA(14,14) =  1.0D0
          MATA(14,16) = -RBG2(3)
          MATA(14,18) =  RBG2(1)
          MATB(14)    =  MJC1 - MJC2
*       Z DIRECTION
          MATA(15,6)  = -1.0D0
          MATA(15,7)  = -RAG1(2)
          MATA(15,8)  =  RAG1(1)
          MATA(15,15) =  1.0D0
          MATA(15,16) =  RBG2(2)
          MATA(15,17) = -RBG2(1)
          MATB(15)    =  MKC1 - MKC2
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(16,11) =  RBG2(3)
          MATA(16,12) = -RBG2(2)
          MATA(16,16) = -IXXT(2)
          MATA(16,17) =  IXYT(2)
          MATA(16,18) =  IXZT(2)
          MATA(16,20) = -RAG2(3)
          MATA(16,21) =  RAG2(2)
          MATB(16)    =  T2X - T1X
*       Y DIRECTION
          MATA(17,10) = -RBG2(3)
          MATA(17,12) =  RBG2(1)
          MATA(17,16) =  IXYT(2)
          MATA(17,17) = -IYYT(2)
          MATA(17,18) =  IYZT(2)
          MATA(17,19) =  RAG2(3)
          MATA(17,21) = -RAG2(1)
          MATB(17)    =  T2Y - T1Y
*       Z DIRECTION
          MATA(18,10) =  RBG2(2)
          MATA(18,11) = -RBG2(1)
          MATA(18,16) =  IXZT(2)
          MATA(18,17) =  IYZT(2)
          MATA(18,18) = -IZZT(2)
          MATA(18,19) = -RAG2(2)
          MATA(18,20) =  RAG2(1)
          MATB(18)    =  T2Z - T1Z
**      LINK THREE
*       SUM OF FORCES
*       X DIRECTION
530       MATA(19,19) =  1.0D0
          MATA(19,22) =  M3
          MATB(19)    =  0.0D0
*       Y DIRECTION
          MATA(20,20) =  1.0D0
          MATA(20,23) =  M3
          MATB(20)    =  0.0D0
*       Z DIRECTION
          MATA(21,21) =  1.0D0
          MATA(21,24) =  M3
```

```
        MATB(21) = -WG3
*       EQUATIONS AT JOINT TWO
*       X DIRECTION
        MATA(22,13) = -1.0D0
        MATA(22,17) = -RAG2(3)
        MATA(22,18) =  RAG2(2)
        MATA(22,22) =  1.0D0
        MATA(22,26) =  RBG3(3)
        MATA(22,27) = -RBG3(2)
        MATB(22)    =  MIC3 - MIC4
*       Y DIRECTION
        MATA(23,14) = -1.0D0
        MATA(23,16) =  RAG2(3)
        MATA(23,18) = -RAG2(1)
        MATA(23,23) =  1.0D0
        MATA(23,25) = -RBG3(3)
        MATA(23,27) =  RBG3(1)
        MATB(23)    =  MJC3 - MJC4
*       Z DIRECTION
        MATA(24,15) = -1.0D0
        MATA(24,16) = -RAG2(2)
        MATA(24,17) =  RAG2(1)
        MATA(24,24) =  1.0D0
        MATA(24,25) =  RBG3(2)
        MATA(24,26) = -RBG3(1)
        MATB(24)    =  MKC3 - MKC4
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
        MATA(25,20) =  RBG3(3)
        MATA(25,21) = -RBG3(2)
        MATA(25,25) = -IXXT(3)
        MATA(25,26) =  IXYT(3)
        MATA(25,27) =  IXZT(3)
        MATB(25)    = -T2X
*       Y DIRECTION
        MATA(26,19) = -RBG3(3)
        MATA(26,21) =  RBG3(1)
        MATA(26,25) =  IXYT(3)
        MATA(26,26) = -IYYT(3)
        MATA(26,27) =  IYZT(3)
        MATB(26)    = -T2Y
*       Z DIRECTION
        MATA(27,19) =  RBG3(2)
        MATA(27,20) = -RBG3(1)
        MATA(27,25) =  IXZT(3)
        MATA(27,26) =  IYZT(3)
        MATA(27,27) = -IZZT(3)
        MATB(27)    = -T2Z

******   FIRST LINK IS CONSTRAINED TO ROTATE IN Z DIRECTION   ******
****    MODIFIED BY RM VERBOS
590       DO 600 I = 4,8
            DO 595 J = 1,27
              MATA(I,J) = 0.0D0
*             MATA(J,I) = 0.0D0
595         CONTINUE
```

```
            MATB(I) = 0.0D0
600       CONTINUE
            MATA(4,4) = 1.0D0
            MATA(5,5) = 1.0D0
            MATA(6,6) = 1.0D0
            MATA(7,7) = 1.0D0
            MATA(8,8) = 1.0D0

605       DO 610 J = 1,27
             MATA(18,J) = 0.0D0
             MATA(27,J) = 0.0D0
610       CONTINUE

            MATA(9,9)   = -(IZZT(1)+IZZT(2)+IZZT(3))
            MATA(18,9)  = -1.0D0
            MATA(18,18) =  1.0D0
            MATB(18)    =  0.0D0
            MATA(27,9)  = -1.0D0
            MATA(27,27) =  1.0D0
            MATB(27)    =  0.0D0

*         CALL EQUATION SOLVER PROGRAM FROM IMSL
620           CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
              IF (IER.EQ.0) THEN
              GOTO 640
              ELSE
              WRITE (*,624) IER
624           FORMAT (I5)
              DO 635  I = 1, 27
              WRITE (*,627) I
627           FORMAT (I7)
              DO 631  J = 1, 27, 3
              WRITE (*,630) J,MATA(I,J),J+1,MATA(I,J+1),J+2,MATA(I,J+2)
630           FORMAT (I5,F13.5,I5,F13.5,I5,F13.5)
631           CONTINUE
              WRITE (*,633) I,MATB(I)
633           FORMAT (I3,F15.5)
635           CONTINUE
              CALL ENDJOB
              END IF

***       FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ

*         JOINT ZERO
640          FX0 = MATB(1)
             FY0 = MATB(2)
             FZ0 = MATB(3)
*         LINK ONE
*         SINCE LINK1 IS CONSTRAIN TO ROTATE AROUND THE Z ONLY
             AX1 = 0.0D0
             AY1 = 0.0D0
             AZ1 = 0.0D0
*660         AX1        = MATB(4)
*            VELX1      = INTGRL(0.,AX1)
*            LCOGX1     = INTGRL(X1,VELX1)
*            LCOGX(1)   = LCOGX1
```

```
*          AY1        = MATB(5)
*          VELY1      = INTGRL(0.,AY1)
*          LCOGY1     = INTGRL(Y1,VELY1)
*          LCOGY(1)   = LCOGY1
*          AZ1        = MATB(6)
*          VELZ1      = INTGRL(0.,AZ1)
*          LCOGZ1     = INTGRL(Z1,VELZ1)
*          LCOGZ(1)   = LCOGZ1
          WD1X       = MATB(7)
          W1X        = INTGRL(0.,WD1X)
          WDX(1)     = WD1X
          W1(1)      = W1X
          WD1Y       = MATB(8)
          W1Y        = INTGRL(0.,WD1Y)
          WDY(1)     = WD1Y
          W1(2)      = W1Y
          WD1Z       = MATB(9)
          W1Z        = INTGRL(0.,WD1Z)
          WDZ(1)     = WD1Z
          W1(3)      = W1Z
****      ADDED BY R.M.VERBOS
685       THZ        = INTGRL(0.,W1Z)
          THZANG     = THZ * RADEG
          COSTHZ     = DCOS(THZ)
          SINTHZ     = DSIN(THZ)

*   IF THE 1ST LINK IS CONSTRAIN TO ROTATE IN THE Z DIRECTION ONLY
*   THE DIRECTION COSINE AND DIRECTION COSINE  ANGLES ARE CONSTANT
*   AND DO NOT NEED TO BE CALCULATED
*
*   CALC DIRECTIONAL COSINES FOR LINK ONE
*
*700        DRCSX(1)  = LCOGX1 / LNCOG1
*          DRCSY(1)  = LCOGY1 / LNCOG1
*          DRCSZ(1)  = LCOGZ1 / LNCOG1
*          AMP = DSQRT(DRCSX(1)*DRCSX(1)+DRCSY(1)*DRCSY(1)+...
*                      DRCSZ(1)*DRCSZ(1))
*          DRCSX(1) = DRCSX(1)/AMP
*          DRCSY(1) = DRCSY(1)/AMP
*          DRCSZ(1) = DRCSZ(1)/AMP
*720       DRCANX(1) = DACOS(DRCSX(1)) * RADEG
*          DRCANY(1) = DACOS(DRCSY(1)) * RADEG
*          DRCANZ(1) = DACOS(DRCSZ(1)) * RADEG
*
**       JOINT LOCATION
*730       JX1  =  LINKL1 * DRCSX(1)
*          JY1  =  LINKL1 * DRCSY(1)
*          JZ1  =  LINKL1 * DRCSZ(1)

*       JOINT ONE
735       FX1 = MATB(10)
          FY1 = MATB(11)
          FZ1 = MATB(12)
**       LINK TWO
740       AX2        = MATB(13)
          VELX2      = INTGRL(0.,AX2)
```

```
          LCOGX2     = INTGRL(X2,VELX2)
          LCOGX(2)   = LCOGX2
          AY2        = MATB(14)
          VELY2      = INTGRL(0.,AY2)
          LCOGY2     = INTGRL(Y2,VELY2)
          LCOGY(2)   = LCOGY2
          AZ2        = MATB(15)
          VELZ2      = INTGRL(0.,AZ2)
          LCOGZ2     = INTGRL(Z2,VELZ2)
          LCOGZ(2)   = LCOGZ2
          WD2X       = MATB(16)
          W2X        = INTGRL(0.,WD2X)
          WDX(2)     = WD2X
          W2(1)      = W2X
          WD2Y       = MATB(17)
          W2Y        = INTGRL(0.,WD2Y)
          WDY(2)     = WD2Y
          W2(2)      = W2Y
          WD2Z       = MATB(18)
          W2Z        = INTGRL(0.,WD2Z)
          WDZ(2)     = WD2Z
          W2(3)      = W2Z

*       GET THE DIRECTION COSINES FOR THE LINK TWO
          DRCSX(2)   = (LCOGX2 - JX1) / LNCOG2
          DRCSY(2)   = (LCOGY2 - JY1) / LNCOG2
          DRCSZ(2)   = (LCOGZ2 - JZ1) / LNCOG2
790       AMP = DSQRT(DRCSX(2)*DRCSX(2)+DRCSY(2)*DRCSY(2)+...
                      DRCSZ(2)*DRCSZ(2))
          DRCSX(2) = DRCSX(2)/AMP
          DRCSY(2) = DRCSY(2)/AMP
          DRCSZ(2) = DRCSZ(2)/AMP
          DRCANX(2) = DACOS(DRCSX(2)) * RADEG
          DRCANY(2) = DACOS(DRCSY(2)) * RADEG
          DRCANZ(2) = DACOS(DRCSZ(2)) * RADEG

*       JOINT LOCATION
800       JX2 = JX1 + LINKL2 * DRCSX(2)
          JY2 = JY1 + LINKL2 * DRCSY(2)
          JZ2 = JZ1 + LINKL2 * DRCSZ(2)

*       JOINT TWO
805       FX2 = MATB(19)
          FY2 = MATB(20)
          FZ2 = MATB(21)
**       LINK THREE
812       AX3        = MATB(22)
          VELX3      = INTGRL(0.,AX3)
          LCOGX3     = INTGRL(X3,VELX3)
          LCOGX(3)   = LCOGX3
          AY3        = MATB(23)
          VELY3      = INTGRL(0.,AY3)
          LCOGY3     = INTGRL(Y3,VELY3)
          LCOGY(3)   = LCOGY3
          AZ3        = MATB(24)
          VELZ3      = INTGRL(0.,AZ3)
```

```
          LCOGZ3    = INTGRL(Z3,VELZ3)
          LCOGZ(3)  = LCOGZ3
          WD3X      = MATB(25)
          W3X       = INTGRL(0.,WD3X)
          WDX(3)    = WD3X
          W3(1)     = W3X
          WD3Y      = MATB(26)
          W3Y       = INTGRL(0.,WD3Y)
          WDY(3)    = WD3Y
          W3(2)     = W3Y
          WD3Z      = MATB(27)
          W3Z       = INTGRL(0.,WD3Z)
          WDZ(3)    = WD3Z
          W3(3)     = W3Z

*  CALC DIRECTIONAL COSINES FOR LINK THREE
845       DRCSX(3) = (LCOGX3 - JX2) / LNCOG3
          DRCSY(3) = (LCOGY3 - JY2) / LNCOG3
          DRCSZ(3) = (LCOGZ3 - JZ2) / LNCOG3
865       AMP = DSQRT(DRCSX(3)*DRCSX(3)+DRCSY(3)*DRCSY(3)+...
                      DRCSZ(3)*DRCSZ(3))
          DRCSX(3) = DRCSX(3)/AMP
          DRCSY(3) = DRCSY(3)/AMP
          DRCSZ(3) = DRCSZ(3)/AMP
          DRCANX(3) = DACOS(DRCSX(3)) * RADEG
          DRCANY(3) = DACOS(DRCSY(3)) * RADEG
          DRCANZ(3) = DACOS(DRCSZ(3)) * RADEG

*   TIP LOCATION
875       TIPX = JX2 + LINKL3 * DRCSX(3)
          TIPY = JY2 + LINKL3 * DRCSY(3)
          TIPZ = JZ2 + LINKL3 * DRCSZ(3)

*         FIND THE ANGLE BETWEEN THE LIMKS
880       ANG23X = DRCANX(2) - DRCANX(3)
          ANG23Y = DRCANY(2) - DRCANY(3)
          ANG23Z = DRCANZ(2) - DRCANZ(3)

          ANG12X = DRCANX(1) - DRCANX(2)
          ANG12Y = DRCANY(1) - DRCANY(2)
          ANG12Z = DRCANZ(1) - DRCANZ(2)


DYNAMIC
NOSORT
******  INPUT TORQUE AT JOINTS
*  CALCULATE THE MAGNITUDE OF THE LENGTH OF THE PROJECTION OF
*  LINK 2 AND 3 IN THE X Y PLANE WHICH IS THE MOMENT ARM FOR
*  THE CALCULATION OF THE GRAVITATIONAL TORQUE

900        MARM2A = DSQRT ( LCOGX2 * LCOGX2 + LCOGY2 * LCOGY2 )
           MARM2B = DSQRT ( LCOGX3 * LCOGX3 + LCOGY3 * LCOGY3 )
           MARM3  = DSQRT ( (LCOGX3 - JX2) * (LCOGX3 - JX2) +...
                            (LCOGY3 - JY2) * (LCOGY3 - JY2) )

*  CALCULATE GRAVITATIONAL TORQUES
```

```
          TG1 = MARM2A * WG2 + MARM2B * WG3
          TG2 = MARM3  * WG3

*  INPUT TORQUES OVERTOP OF GRAVITY
*10       T1FNC =  2*A - A * DCOS ( W * TIME )
*         T2FNC =  -3 * A * DSIN (W*TIME)
*         T0Z   =  A * DSIN ( W * TIME )
*         T0Z   =  A - A * DSIN ( W * TIME )
*         T0Z   =  50.0 - 50.0 * TIME

*  CALCULATE TOTAL TORQUE ON JOINT 1 AND 2
           T1 = TG1 + T1FNC
           T2 = TG2 + T2FNC

*  CALCULATE X AND Y COMPONENT OF TORQUE ON JOINT 1 AND 2
           T1X = T1 * DCOS (THZ)
           T1Y = T1 * DSIN (THZ)
           T2X = T2 * COS (THZ)
           T2Y = T2 * SIN (THZ)

*     TORQUE CHECK
           T1CH = DSQRT ( T1X * T1X + T1Y * T1Y )
           T2CH = DSQRT ( T2X * T2X + T2Y * T2Y )

END
STOP

FORTRAN

*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS

         SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
940         IMPLICIT REAL*8 (A-Z)
            DIMENSION VECTA(3),VECTB(3)
            MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
            MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
            MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
          RETURN

          END
```

APPENDIX B

DOUBLE PENDULUM VALIDATION PROGRAM

```
TITLE DOUBLE PENDULUM VALIDATION PROGRAM

*****************************************************************************
*  THIS IS A PROGRAM THAT WILL SOLVE THE DIRECT DYNAMICS PROBLEM FOR A      *
*  2 LINK RIGID BODY MANIPULATOR WITHOUT THE USE OF THE STANDARD TRAN-      *
*  FORMATION MATRICES AND VALIDATE THE MOTION BY THE USE OF THE DOUBLE      *
*  PENDULUM PROBLEM.                                                        *
*                                LT  R. M.  VERBOS   USN                    *
*****************************************************************************

TERMINAL
METHOD ADAMS
PRINT .10,ANGTH2,TH2ANG,ANGTH3,TH3ANG,ANG23
CONTROL FINTIM =4.0, DELT = .0005, DELMAX = .1, DELPRT = .10
SAVE .05, ANGTH2,TH2ANG,ANGTH3,TH3ANG,ANG23
GRAPH (DE=TEK618) TIME,ANGTH2,TH2ANG
GRAPH (DE=TEK618) TIME,ANGTH3,TH3ANG
GRAPH (DE=TEK618) TIME,ANG23
D     DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D     DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D     DIMENSION IMAT(3,3,3),NIMAT(3,3),TMAT(3,3),TMATTR(3,3),MATTMP(3,3)
D     DIMENSION LIMAT(3,3,3)
FIXED   IER,I,J,M,K,P,N,IA,IDGT,A,I1
ARRAY MATB(27),ABMATB(27),LCOGX(3),LCOGY(3),LCOGZ(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY VECTA(3),VECTB(3)
ARRAY WDX(3),WDY(3),WDZ(3),W1(3),W2(3),W3(3)
ARRAY RBG1(3),RAG1(3),RBG2(3),RAG2(3),RBG3(3)
ARRAY WKAREA(850)
ARRAY IXXT(3),IYYT(3),IZZT(3),IXYT(3),IXZT(3),IYZT(3)
ARRAY DRCANX(3),DRCANY(3),DRCANZ(3)
ARRAY DRCSX(3),DRCSY(3),DRCSZ(3)

INITIAL

******   INPUT   ******
*       INPUT PARAMETER CONSTANTS
55          A = 15.0D0
            P = 0.0D0
            W =  2 * PI
            IDGT = 6
***     MODIFIED BY  RM VERBOS
           G = 9.81D0
*          G = 0.0D0
           N = 27
```

```
          M = 1
          IA = 27
          IER = 0
          LEVELQ = 0
          LEVLDQ = 0

*       INPUT JOINT LOCATIONS IN METERS
          JX0 = 0.0D0
          JY0 = 0.0D0
          JZ0 = 0.0D0
          JX1 = 0.0D0
          JY1 = 0.0D0
          JZ1 = 0.2D0
          JX2 = 0.0D0
          JY2 = 0.416D0
          JZ2 = 0.2D0

*       INPUT TORQUE CONSTANTS
          T0X    = 0.0D0
          T0Y    = 0.0D0
          T0Z    = 0.0D0
          T1X    = 0.0D0
          T1Y    = 0.0D0
          T1Z    = 0.0D0
          T2X    = 0.0D0
          T2Y    = 0.0D0
          T2Z    = 0.0D0

*       INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS
*       FOR EACH LINK ENDS
          L(1,1) = 0.05D0
          L(1,2) = 0.15D0
          L(2,1) = 0.20D0
          L(2,2) = 0.216D0
          L(3,1) = 0.225D0
          L(3,2) = 0.226D0

* INPUT THE LINK LENGTHS OF THE ROBOT
          LINKL1 = 0.20D0
          LINKL2 = 0.416D0
          LINKL3 = 0.451D0

*       INPUT MASS AT LINK ENDS IN KILOGRAMS
110       MASS(1,1) = 11.0D0
          MASS(1,2) = 33.0D0
          MASS(2,1) = 2.2D0
          MASS(2,2) = 2.2D0
          MASS(3,1) = 28.6D0
          MASS(3,2) = 28.6D0

*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
120        LCOGX(1) =  0.0D0
           X1       =  LCOGX(1)
           LCOGY(1) =  0.0D0
           Y1       =  LCOGY(1)
           LCOGZ(1) =  0.10D0
```

```
            Z1       =  LCOGZ(1)
            LCOGX(2) =  0.0D0
            X2       =  LCOGX(2)
            LCOGY(2) =  0.208D0
            Y2       =  LCOGY(2)
            LCOGZ(2) =  0.20D0
            Z2       =  LCOGZ(2)
            LCOGX(3) =  0.0D0
            X3       =  LCOGX(3)
            LCOGY(3) =  0.6415D0
            Y3       =  LCOGY(3)
            LCOGZ(3) =  0.2D0
            Z3       =  LCOGZ(3)

*       INPUT MASS OF EACH LINK IN KG
            M1 = 44.0D0
            M2 = 4.4D0
            M3 = 57.2D0

*       INPUT ACCELERATIONS OF JOINT ZERO
            AX0 = 0.0D0
            AY0 = 0.0D0
            AZ0 = 0.0D0

*       INPUT THE INITIAL DIRECTION COSINES
            DRCSX(1) = 0.0D0
            DRCSY(1) = 0.0D0
            DRCSZ(1) = 1.0D0
            DRCSX(2) = 0.0D0
            DRCSY(2) = 1.0D0
            DRCSZ(2) = 0.0D0
            DRCSX(3) = 0.0D0
            DRCSY(3) = 1.0D0
            DRCSZ(3) = 0.0D0

*       INPUT  THE INITIAL DIRECTION COSINE ANGLES
            DRCANX(1) = 90.0D0
            DRCANY(1) = 90.0D0
            DRCANZ(1) =  0.0D0
            DRCANX(2) = 90.0D0
            DRCANY(2) =  0.0D0
            DRCANZ(2) = 90.0D0
            DRCANX(3) = 90.0D0
            DRCANY(3) =  0.0D0
            DRCANZ(3) = 90.0D0

******   INITIALIZE   ******
*       OMEGA AND OMEGA DOT
160         DO 170 I = 1,3
               W1(I)     = 0.0D0
               W2(I)     = 0.0D0
               W3(I)     = 0.0D0
               WDX(I)    = 0.0D0
               WDY(I)    = 0.0D0
               WDZ(I)    = 0.0D0
170         CONTINUE
```

```
          THZ = 0.0D0

*      INITIALIZE MATRIX A AND B TO ZERO
           DO 180 I = 1,27
              DO 175 J = 1,27
                 MATA(I,J) = 0.0D0
175           CONTINUE
               MATB(I) = 0.0D0
180        CONTINUE

******    CALCULATIONS    ******
*       WEIGHTS (NEWTONS)
185        WG1 = M1*G
           WG2 = M2*G
           WG3 = M3*G

*       COMPUTE THE LENGTH FROM THE INBOARD JOINT TO COG
           LNCOG1 = DSQRT ( LCOGX(1)*LCOGX(1) + LCOGY(1)*LCOGY(1) +...
                            LCOGZ(1)*LCOGZ(1) )
           LX2 = LCOGX(2) - JX1
           LY2 = LCOGY(2) - JY1
           LZ2 = LCOGZ(2) - JZ1
           LNCOG2 = DSQRT ( LX2*LX2 + LY2*LY2 + LZ2*LZ2 )
           LX3 = LCOGX(3) - JX2
           LY3 = LCOGY(3) - JY2
           LZ3 = LCOGZ(3) - JZ2
           LNCOG3 = DSQRT ( LX3*LX3 + LY3*LY3 + LZ3*LZ3 )

*       COMPUTE INITIAL INERTIAS BASED ON POINT MASSES
*                IN GLOBAL COORDINATES
190        DO 225 I = 1,3
              RX(I,1) = -L(I,1) * DRCSX(I)
              RX(I,2) =  L(I,2) * DRCSX(I)
              RY(I,1) = -L(I,1) * DRCSY(I)
              RY(I,2) =  L(I,2) * DRCSY(I)
              RZ(I,1) = -L(I,1) * DRCSZ(I)
              RZ(I,2) =  L(I,2) * DRCSZ(I)
200    IXX(I,1) = MASS(I,1) * ((RY(I,1) * RY(I,1)) + (RZ(I,1) * RZ(I,1)))
       IXX(I,2) = MASS(I,2) * ((RY(I,2) * RY(I,2)) + (RZ(I,2) * RZ(I,2)))
       IXXT(I)  = IXX(I,1) + IXX(I,2)
       IYY(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RZ(I,1) * RZ(I,1)))
       IYY(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RZ(I,2) * RZ(I,2)))
       IYYT(I)  = IYY(I,1) + IYY(I,2)
       IZZ(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RY(I,1) * RY(I,1)))
       IZZ(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RY(I,2) * RY(I,2)))
       IZZT(I)  = IZZ(I,1) + IZZ(I,2)
205    IXY(I,1) = MASS(I,1) * RX(I,1) * RY(I,1)
       IXY(I,2) = MASS(I,2) * RX(I,2) * RY(I,2)
       IXYT(I)  = IXY(I,1) + IXY(I,2)
       IXZ(I,1) = MASS(I,1) * RZ(I,1) * RX(I,1)
       IXZ(I,2) = MASS(I,2) * RZ(I,2) * RX(I,2)
       IXZT(I)  = IXZ(I,1) + IXZ(I,2)
       IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
       IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
       IYZT(I)  = IYZ(I,1) + IYZ(I,2)
```

```
      IF (IXXT(I) .LE. .01) THEN
      IXXT(I)  = .01
      ELSE
      IXXT(I) = IXXT(I)
      END IF
      IF (IYYT(I) .LE. .02) THEN
      IYYT(I)  = .02
      ELSE
      IYYT(I)  = IYYT(I)
      END IF
      IF (IZZT(I) .LE. .02) THEN
      IZZT(I)  = .02
      ELSE
      IZZT(I)  = IZZT(I)
      END IF
           IMAT(I,1,1) =  IXXT(I)
           IMAT(I,1,2) =  IXYT(I)
           IMAT(I,1,3) =  IXZT(I)
           IMAT(I,2,1) = -IXYT(I)
           IMAT(I,2,2) =  IYYT(I)
           IMAT(I,2,3) =  IYZT(I)
           IMAT(I,3,1) = -IXZT(I)
           IMAT(I,3,2) = -IYZT(I)
           IMAT(I,3,3) =  IZZT(I)
225        CONTINUE

*      DUE TO LINK 1 CONSTRAINTS LINK 1 INERTIAS ARE CONSTANT
         IXXT(1) = IMAT(1,1,1)
         IXYT(1) = IMAT(1,1,2)
         IXZT(1) = IMAT(1,1,3)
         IYYT(1) = IMAT(1,2,2)
         IYZT(1) = IMAT(1,2,3)
         IZZT(1) = IMAT(1,3,3)

*    TRANSFORM THE INITIAL INERTIAS TO LOCAL COORDINATED
         DO 9 I = 2, 3
           TMAT(2,1) = -DCOS ( THZ )
           TMAT(2,2) = -DSIN ( THZ )
           TMAT(2,3) = 0.0D0
           TMAT(3,1) = DRCSX(I)
           TMAT(3,2) = DRCSY(I)
           TMAT(3,3) = DRCSZ(I)

           VECTA(1)  = TMAT(2,1)
           VECTA(2)  = TMAT(2,2)
           VECTA(3)  = TMAT(2,3)
           VECTB(1)  = TMAT(3,1)
           VECTB(2)  = TMAT(3,2)
           VECTB(3)  = TMAT(3,3)
           CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
           TMAT(1,1) = MI1
           TMAT(1,2) = MJ1
           TMAT(1,3) = MK1
           TMATTR(1,1) = TMAT(1,1)
           TMATTR(1,2) = TMAT(2,1)
           TMATTR(1,3) = TMAT(3,1)
```

```
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)

            DO 5 I1 = 1,3
            DO 5 J  = 1,3
               TEMP = 0.0D0
              DO 1  K  = 1,3
                 TEMP = TMAT(I1,K) * IMAT(I,K,J) + TEMP
1             CONTINUE
               MATTMP(I1,J) = TEMP
5           CONTINUE

            DO 8   I1 = 1,3
            DO 8   J  = 1,3
               TEMP = 0.0D0
              DO 7   K  = 1,3
                 TEMP = MATTMP(I1,K) * TMATTR(K,J) + TEMP
7             CONTINUE
             LIMAT(I,I1,J) = TEMP
8           CONTINUE
9         CONTINUE

*     DOUBLE PENDULUM INITIALIZATION
        TH2D0 = 0.0D0
        TH3D0 = 0.0D0
        TH30  = PI/2.0
        TH20  = PI/2.0
        L2    = 0.416D0
        L3    = 0.451D0
        M2P   = 30.8D0
        M3P   = 28.6D0
        HF    = 0.5D0

DERIVATIVE
NOSORT
230         CALL ERRSET (208,256,-1,1,1)
            CALL UERSET(LEVELQ,LEVLDQ)


*      INITIALIZE MATRIX A AND B TO ZERO
            DO 240 I = 1,27
               DO 235 J = 1,27
                  MATA(I,J) = 0.0D0
235           CONTINUE
               MATB(I) = 0.0D0
240         CONTINUE

*       INPUT JOINT EQUATIONS

*       JOINT ZERO  EQUATIONS
*       W1 X (W1 X RB/G1)
*             RBG1(1) = JX0 - LCOGX(1)
```

```
*              RBG1(2) = JY0 - LCOGY(1)
*              RBG1(3) = JZ0 - LCOGZ(1)
*              VECTA(1) = W1(1)
*              VECTA(2) = W1(2)
*              VECTA(3) = W1(3)
*           CALL CPROD(VECTA0,RBG1,MIC0,MJC0,MKC0)
*              VECTB0(1) = MIC0
*              VECTB0(2) = MJC0
*              VECTB0(3) = MKC0
*           CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)
**       SINCE LINK ONE DOES NOT MOVE
           MIC0 = 0.0D0
           MJC0 = 0.0D0
           MKC0 = 0.0D0

*       INPUT JOINT EQUATIONS
*       JOINT ZERO  EQUATIONS
*        W1 X (W1 X RB/G1)
              RBG1(1)  = JX0 - LCOGX(1)
              RBG1(2)  = JY0 - LCOGY(1)
              RBG1(3)  = JZ0 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD(VECTA,RBG1,MIC0,MJC0,MKC0)
              VECTB(1) = MIC0
              VECTB(2) = MJC0
              VECTB(3) = MKC0
           CALL CPROD(VECTA,VECTB,MIC0,MJC0,MKC0)
*       W1 X (W1 X RA/G1)
              RAG1(1) = JX1 - LCOGX(1)
              RAG1(2) = JY1 - LCOGY(1)
              RAG1(3) = JZ1 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD (VECTA,RAG1,MIC1,MJC1,MKC1)
              VECTB(1) = MIC1
              VECTB(2) = MJC1
              VECTB(3) = MKC1
           CALL CPROD (VECTA,VECTB,MIC1,MJC1,MKC1)
*       W2 X (W2 X RB/G2)
              RBG2(1) = JX1 - LCOGX(2)
              RBG2(2) = JY1 - LCOGY(2)
              RBG2(3) = JZ1 - LCOGZ(2)
              VECTA(1) = W2(1)
              VECTA(2) = W2(2)
              VECTA(3) = W2(3)
           CALL CPROD (VECTA,RBG2,MIC2,MJC2,MKC2)
              VECTB(1) = MIC2
              VECTB(2) = MJC2
              VECTB(3) = MKC2
           CALL CPROD (VECTA,VECTB,MIC2,MJC2,MKC2)
*       W2 X (W2 X RA/G2)
              RAG2(1) = JX2 - LCOGX(2)
              RAG2(2) = JY2 - LCOGY(2)
```

```
            RAG2(3) = JZ2 - LCOGZ(2)
            VECTA(1) = W2(1)
            VECTA(2) = W2(2)
            VECTA(3) = W2(3)
          CALL CPROD (VECTA,RAG2,MIC3,MJC3,MKC3)
            VECTB(1) = MIC3
            VECTB(2) = MJC3
            VECTB(3) = MKC3
          CALL CPROD(VECTA,VECTB,MIC3,MJC3,MKC3)
*       W3 X (W3 X RB/G3)
            RBG3(1) = JX2 - LCOGX(3)
            RBG3(2) = JY2 - LCOGY(3)
            RBG3(3) = JZ2 - LCOGZ(3)
            VECTA(1) = W3(1)
            VECTA(2) = W3(2)
            VECTA(3) = W3(3)
          CALL CPROD (VECTA,RBG3,MIC4,MJC4,MKC4)
            VECTB(1) = MIC4
            VECTB(2) = MJC4
            VECTB(3) = MKC4
          CALL CPROD (VECTA,VECTB,MIC4,MJC4,MKC4)

*       INERTIA TRANSFORMATION
          DO 390 I = 2, 3
            TMAT(2,1) = -DCOS ( THZ )
            TMAT(2,2) = -DSIN ( THZ )
            TMAT(2,3) = 0.0D0
            TMAT(3,1) = DRCSX(I)
            TMAT(3,2) = DRCSY(I)
            TMAT(3,3) = DRCSZ(I)

            VECTA(1)  = TMAT(2,1)
            VECTA(2)  = TMAT(2,2)
            VECTA(3)  = TMAT(2,3)
            VECTB(1)  = TMAT(3,1)
            VECTB(2)  = TMAT(3,2)
            VECTB(3)  = TMAT(3,3)
            CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
            TMAT(1,1) = MI1
            TMAT(1,2) = MJ1
            TMAT(1,3) = MK1
            TMATTR(1,1) = TMAT(1,1)
            TMATTR(1,2) = TMAT(2,1)
            TMATTR(1,3) = TMAT(3,1)
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)

            DO 375 I1 = 1,3
            DO 375 J  = 1,3
               TEMP = 0.0D0
              DO 370  K  = 1,3
```

```
                TEMP = TMATTR(I1,K) * LIMAT(I,K,J) + TEMP
370           CONTINUE
               MATTMP(I1,J) = TEMP
375         CONTINUE

            DO 380 I1 = 1,3
            DO 380 J  = 1,3
               TEMP = 0.0D0
              DO 378 K  = 1,3
                 TEMP = MATTMP(I1,K) * TMAT(K,J) + TEMP
378           CONTINUE
             NIMAT(I1,J) = TEMP
380         CONTINUE

             IXXT(I) = NIMAT(1,1)
             IXYT(I) = NIMAT(1,2)
             IXZT(I) = NIMAT(1,3)
             IYYT(I) = NIMAT(2,2)
             IYZT(I) = NIMAT(2,3)
             IZZT(I) = NIMAT(3,3)

390     CONTINUE

******  ENTER CONSTANTS INTO MATRIX A  ******

***     LINK ONE
*       SUM OF FORCES
*       X DIRECTION
395       MATA(1,1)  =  1.0D0
          MATA(1,4)  =  M1
          MATA(1,10) = -1.0D0
          MATB(1)    =  0.0D0
*       Y DIRECTION
          MATA(2,2)  =  1.0D0
          MATA(2,5)  =  M1
          MATA(2,11) = -1.0D0
          MATB(2)    =  0.0D0
*       Z DIRECTION
          MATA(3,3)  =  1.0D0
          MATA(3,6)  =  M1
          MATA(3,12) = -1.0D0
          MATB(3)    = -WG1
*       EQUATIONS AT JOINT ZERO
*       X DIRECTION
          MATA(4,4) =  1.0D0
          MATA(4,8) =  RBG1(3)
          MATA(4,9) = -RBG1(2)
          MATB(4)   =  AX0 - MIC0
*       Y DIRECTION
          MATA(5,5) =  1.0D0
          MATA(5,7) = -RBG1(3)
          MATA(5,9) =  RBG1(1)
          MATB(5)   =  AY0 - MJC0
*       Z DIRECTION
          MATA(6,6) =  1.0D0
          MATA(6,7) =  RBG1(2)
```

```
          MATA(6,8) = -RBG1(1)
          MATB(6)   =  AZ0 - MKC0
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(7,2)  =  RBG1(3)
          MATA(7,3)  = -RBG1(2)
          MATA(7,7)  = -IXXT(1)
          MATA(7,8)  =  IXYT(1)
          MATA(7,9)  =  IXZT(1)
          MATA(7,11) = -RAG1(3)
          MATA(7,12) =  RAG1(2)
          MATB(7)    =  T1X - T0X
*       Y DIRECTION
          MATA(8,1)  = -RBG1(3)
          MATA(8,3)  =  RBG1(1)
          MATA(8,7)  =  IXYT(1)
          MATA(8,8)  = -IYYT(1)
          MATA(8,9)  =  IYZT(1)
          MATA(8,10) =  RAG1(3)
          MATA(8,12) = -RAG1(1)
          MATB(8)    =  T1Y - T0Y
*       Z DIRECTION
          MATA(9,1)  =  RBG1(2)
          MATA(9,2)  = -RBG1(1)
          MATA(9,7)  =  IXZT(1)
          MATA(9,8)  =  IYZT(1)
          MATA(9,9)  = -IZZT(1)
          MATA(9,10) = -RAG1(2)
          MATA(9,11) =  RAG1(1)
          MATB(9)    =  T1Z - T0Z

**      LINK TWO
*       SUM OF FORCES
*       X DIRECTION
460       MATA(10,10) =  1.0D0
          MATA(10,13) =  M2
          MATA(10,19) = -1.0D0
          MATB(10)    =  0.0D0
*       Y DIRECTION
          MATA(11,11) =  1.0D0
          MATA(11,14) =  M2
          MATA(11,20) = -1.0D0
          MATB(11)    =  0.0D0
*       Z DIRECTION
          MATA(12,12) =  1.0D0
          MATA(12,15) =  M2
          MATA(12,21) = -1.0D0
          MATB(12)    = -WG2
*       EQUATIONS AT JOINT ONE
*       X DIRECTION
          MATA(13,4)  = -1.0D0
          MATA(13,8)  = -RAG1(3)
          MATA(13,9)  =  RAG1(2)
          MATA(13,13) =  1.0D0
          MATA(13,17) =  RBG2(3)
          MATA(13,18) = -RBG2(2)
```

```
        MATB(13)    =  MIC1 - MIC2
*       Y DIRECTION
        MATA(14,5)  = -1.0D0
        MATA(14,7)  =  RAG1(3)
        MATA(14,9)  = -RAG1(1)
        MATA(14,14) =  1.0D0
        MATA(14,16) = -RBG2(3)
        MATA(14,18) =  RBG2(1)
        MATB(14)    =  MJC1 - MJC2
*       Z DIRECTION
        MATA(15,6)  = -1.0D0
        MATA(15,7)  = -RAG1(2)
        MATA(15,8)  =  RAG1(1)
        MATA(15,15) =  1.0D0
        MATA(15,16) =  RBG2(2)
        MATA(15,17) = -RBG2(1)
        MATB(15)    =  MKC1 - MKC2
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
        MATA(16,11) =  RBG2(3)
        MATA(16,12) = -RBG2(2)
        MATA(16,16) = -IXXT(2)
        MATA(16,17) =  IXYT(2)
        MATA(16,18) =  IXZT(2)
        MATA(16,20) = -RAG2(3)
        MATA(16,21) =  RAG2(2)
        MATB(16)    =  T2X - T1X
*       Y DIRECTION
        MATA(17,10) = -RBG2(3)
        MATA(17,12) =  RBG2(1)
        MATA(17,16) =  IXYT(2)
        MATA(17,17) = -IYYT(2)
        MATA(17,18) =  IYZT(2)
        MATA(17,19) =  RAG2(3)
        MATA(17,21) = -RAG2(1)
        MATB(17)    =  T2Y - T1Y
*       Z DIRECTION
        MATA(18,10) =  RBG2(2)
        MATA(18,11) = -RBG2(1)
        MATA(18,16) =  IXZT(2)
        MATA(18,17) =  IYZT(2)
        MATA(18,18) = -IZZT(2)
        MATA(18,19) = -RAG2(2)
        MATA(18,20) =  RAG2(1)
        MATB(18)    =  T2Z - T1Z

**      LINK THREE
*       SUM OF FORCES
*       X DIRECTION
530     MATA(19,19) =  1.0D0
        MATA(19,22) =  M3
        MATB(19)    =  0.0D0
*       Y DIRECTION
        MATA(20,20) =  1.0D0
        MATA(20,23) =  M3
```

```
          MATB(20)    =  0.0D0
*       Z DIRECTION
          MATA(21,21) =  1.0D0
          MATA(21,24) =  M3
          MATB(21) = -WG3
*       EQUATIONS AT JOINT TWO
*       X DIRECTION
          MATA(22,13) = -1.0D0
          MATA(22,17) = -RAG2(3)
          MATA(22,18) =  RAG2(2)
          MATA(22,22) =  1.0D0
          MATA(22,26) =  RBG3(3)
          MATA(22,27) = -RBG3(2)
          MATB(22)    =  MIC3 - MIC4
*       Y DIRECTION
          MATA(23,14) = -1.0D0
          MATA(23,16) =  RAG2(3)
          MATA(23,18) = -RAG2(1)
          MATA(23,23) =  1.0D0
          MATA(23,25) = -RBG3(3)
          MATA(23,27) =  RBG3(1)
          MATB(23)    =  MJC3 - MJC4
*       Z DIRECTION
          MATA(24,15) = -1.0D0
          MATA(24,16) = -RAG2(2)
          MATA(24,17) =  RAG2(1)
          MATA(24,24) =  1.0D0
          MATA(24,25) =  RBG3(2)
          MATA(24,26) = -RBG3(1)
          MATB(24)    =  MKC3 - MKC4
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(25,20) =  RBG3(3)
          MATA(25,21) = -RBG3(2)
          MATA(25,25) = -IXXT(3)
          MATA(25,26) =  IXYT(3)
          MATA(25,27) =  IXZT(3)
          MATB(25)    = -T2X
*       Y DIRECTION
          MATA(26,19) = -RBG3(3)
          MATA(26,21) =  RBG3(1)
          MATA(26,25) =  IXYT(3)
          MATA(26,26) = -IYYT(3)
          MATA(26,27) =  IYZT(3)
          MATB(26)    = -T2Y
*       Z DIRECTION
          MATA(27,19) =  RBG3(2)
          MATA(27,20) = -RBG3(1)
          MATA(27,25) =  IXZT(3)
          MATA(27,26) =  IYZT(3)
          MATA(27,27) = -IZZT(3)
          MATB(27)    = -T2Z

******    FIRST LINK IS CONSTRAINED TO ROTATE IN Z DIRECTION    ******
590       DO 600 I = 4,8
            DO 595 J = 1,27
```

```
            MATA(I,J) = 0.0D0
*           MATA(J,I) = 0.0D0
595        CONTINUE
            MATB(I) = 0.0D0
600      CONTINUE
            MATA(4,4) = 1.0D0
            MATA(5,5) = 1.0D0
            MATA(6,6) = 1.0D0
            MATA(7,7) = 1.0D0
            MATA(8,8) = 1.0D0

605      DO 610 J = 1,27
            MATA(18,J) = 0.0D0
            MATA(27,J) = 0.0D0
610      CONTINUE

           MATA(9,9)   = -(IZZT(1)+IZZT(2)+IZZT(3))
           MATA(18,9)  = -1.0D0
           MATA(18,18) =  1.0D0
           MATB(18)    =  0.0D0
           MATA(27,9)  = -1.0D0
           MATA(27,27) =  1.0D0
           MATB(27)    =  0.0D0


*       CALL EQUATION SOLVER PROGRAM FROM IMSL
620        CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
           IF (IER.EQ.0) THEN
           GOTO 640
           ELSE
           WRITE (*,624) IER
624        FORMAT (I5)
           DO 635  I = 1, 27
           WRITE (*,627) I
627        FORMAT (I7)
           DO 631  J = 1, 27, 3
           WRITE (*,630) J,MATA(I,J),J+1,MATA(I,J+1),J+2,MATA(I,J+2)
630        FORMAT (I5,F13.5,I5,F13.5,I5,F13.5)
631        CONTINUE
           WRITE (*,633) I,MATB(I)
633        FORMAT (I3,F15.5)
635        CONTINUE
           CALL ENDJOB
           END IF

***     FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
***     MODIFIED BY R.M.VERBOS

*       JOINT ZERO
640        FX0 = MATB(1)
           FY0 = MATB(2)
           FZ0 = MATB(3)

*       LINK ONE
*       SINCE LINK1 IS CONSTRAIN NOT TO MOVE ALL ACC AND VEL ARE ZERO
           AX1 = 0.0D0
```

```
          AY1 = 0.0D0
          AZ1 = 0.0D0
*660      AX1       = MATB(4)
*         VELX1     = INTGRL(0.,AX1)
*         LCOGX1    = INTGRL(X1,VELX1)
*         LCOGX(1)  = LCOGX1
*         AY1       = MATB(5)
*         VELY1     = INTGRL(0.,AY1)
*         LCOGY1    = INTGRL(Y1,VELY1)
*         LCOGY(1)  = LCOGY1
*         AZ1       = MATB(6)
*         VELZ1     = INTGRL(0.,AZ1)
*         LCOGZ1    = INTGRL(Z1,VELZ1)
*         LCOGZ(1)  = LCOGZ1
*        WD1X      = MATB(7)
*        W1X       = INTGRL(0.,WD1X)
*        WDX(1)    = WD1X
*        W1(1)     = W1X
*        WD1Y      = MATB(8)
*        W1Y       = INTGRL(0.,WD1Y)
*        WDY(1)    = WD1Y
*        W1(2)     = W1Y
*        WD1Z      = MATB(9)
*        W1Z       = INTGRL(0.,WD1Z)
*        WDZ(1)    = WD1Z
*        W1(3)     = W1Z
***      ADDED BY R.M.VERBOS
685      THZ       = INTGRL(0.,W1Z)
         THZANG    = THZ * RADEG
         COSTHZ    = DCOS(THZ)
         SINTHZ    = DSIN(THZ)

*   IF THE 1ST LINK IS CONSTRAIN TO ROTATE IN THE Z DIRECTION ONLY
*   THE DIRECTION COSINE AND DIRECTION COSINE  ANGLES ARE CONSTANT
*   AND DO NOT NEED TO BE CALCULATED
*
*   CALC DIRECTIONAL COSINES FOR LINK ONE
*
*700       DRCSX(1)  = LCOGX1 / LNCOG1
*         DRCSY(1)  = LCOGY1 / LNCOG1
*         DRCSZ(1)  = LCOGZ1 / LNCOG1
*        AMP = DSQRT(DRCSX(1)*DRCSX(1)+DRCSY(1)*DRCSY(1)+...
*                    DRCSZ(1)*DRCSZ(1))
*        DRCSX(1) = DRCSX(1)/AMP
*        DRCSY(1) = DRCSY(1)/AMP
*        DRCSZ(1) = DRCSZ(1)/AMP
*720     DRCANX(1) = DACOS(DRCSX(1)) * RADEG
*        DRCANY(1) = DACOS(DRCSY(1)) * RADEG
*        DRCANZ(1) = DACOS(DRCSZ(1)) * RADEG
*
**      JOINT LOCATION
*730     JX1  =  LINKL1 * DRCSX(1)
*        JY1  =  LINKL1 * DRCSY(1)
*        JZ1  =  LINKL1 * DRCSZ(1)
```

```
*       JOINT ONE
735       FX1 = MATB(10)
          FY1 = MATB(11)
          FZ1 = MATB(12)
**      LINK TWO
740       AX2        = MATB(13)
          VELX2      = INTGRL(0.,AX2)
          LCOGX2     = INTGRL(X2,VELX2)
          LCOGX(2)   = LCOGX2
          AY2        = MATB(14)
          VELY2      = INTGRL(0.,AY2)
          LCOGY2     = INTGRL(Y2,VELY2)
          LCOGY(2)   = LCOGY2
          AZ2        = MATB(15)
          VELZ2      = INTGRL(0.,AZ2)
          LCOGZ2     = INTGRL(Z2,VELZ2)
          LCOGZ(2)   = LCOGZ2
          WD2X       = MATB(16)
          W2X        = INTGRL(0.,WD2X)
          WDX(2)     = WD2X
          W2(1)      = W2X
          WD2Y       = MATB(17)
          W2Y        = INTGRL(0.,WD2Y)
          WDY(2)     = WD2Y
          W2(2)      = W2Y
          WD2Z       = MATB(18)
          W2Z        = INTGRL(0.,WD2Z)
          WDZ(2)     = WD2Z
          W2(3)      = W2Z

*       GET THE DIRECTION COSINES FOR THE LINK TWO
          DRCSX(2)   = (LCOGX2 - JX1) / LNCOG2
          DRCSY(2)   = (LCOGY2 - JY1) / LNCOG2
          DRCSZ(2)   = (LCOGZ2 - JZ1) / LNCOG2
790       AMP = DSQRT(DRCSX(2)*DRCSX(2)+DRCSY(2)*DRCSY(2)+...
                      DRCSZ(2)*DRCSZ(2))
          DRCSX(2) = DRCSX(2)/AMP
          DRCSY(2) = DRCSY(2)/AMP
          DRCSZ(2) = DRCSZ(2)/AMP
          DRCANX(2) = DACOS(DRCSX(2)) * RADEG
          DRCANY(2) = DACOS(DRCSY(2)) * RADEG
          DRCANZ(2) = DACOS(DRCSZ(2)) * RADEG

*       JOINT LOCATION
800       JX2 = JX1 + LINKL2 * DRCSX(2)
          JY2 = JY1 + LINKL2 * DRCSY(2)
          JZ2 = JZ1 + LINKL2 * DRCSZ(2)

*       JOINT TWO
805       FX2 = MATB(19)
          FY2 = MATB(20)
          FZ2 = MATB(21)

**      LINK THREE
812       AX3        = MATB(22)
          VELX3      = INTGRL(0.,AX3)
```

```
          LCOGX3    = INTGRL(X3,VELX3)
          LCOGX(3)  = LCOGX3
          AY3       = MATB(23)
          VELY3     = INTGRL(0.,AY3)
          LCOGY3    = INTGRL(Y3,VELY3)
          LCOGY(3)  = LCOGY3
          AZ3       = MATB(24)
          VELZ3     = INTGRL(0.,AZ3)
          LCOGZ3    = INTGRL(Z3,VELZ3)
          LCOGZ(3)  = LCOGZ3
          WD3X      = MATB(25)
          W3X       = INTGRL(0.,WD3X)
          WDX(3)    = WD3X
          W3(1)     = W3X
          WD3Y      = MATB(26)
          W3Y       = INTGRL(0.,WD3Y)
          WDY(3)    = WD3Y
          W3(2)     = W3Y
          WD3Z      = MATB(27)
          W3Z       = INTGRL(0.,WD3Z)
          WDZ(3)    = WD3Z
          W3(3)     = W3Z

*  CALC DIRECTIONAL COSINES FOR LINK THREE
845       DRCSX(3) = (LCOGX3 - JX2) / LNCOG3
          DRCSY(3) = (LCOGY3 - JY2) / LNCOG3
          DRCSZ(3) = (LCOGZ3 - JZ2) / LNCOG3
865       AMP = DSQRT(DRCSX(3)*DRCSX(3)+DRCSY(3)*DRCSY(3)+...
                      DRCSZ(3)*DRCSZ(3))
          DRCSX(3) = DRCSX(3)/AMP
          DRCSY(3) = DRCSY(3)/AMP
          DRCSZ(3) = DRCSZ(3)/AMP
          DRCANX(3) = DACOS(DRCSX(3)) * RADEG
          DRCANY(3) = DACOS(DRCSY(3)) * RADEG
          DRCANZ(3) = DACOS(DRCSZ(3)) * RADEG

*   TIP LOCATION
*875       TIPX = JX2 + LINKL3 * DRCSX(3)
*          TIPY = JY2 + LINKL3 * DRCSY(3)
*          TIPZ = JZ2 + LINKL3 * DRCSZ(3)

*         FIND THE ANGLE BETWEEN THE LIMKS
880       ANG23X = DRCANX(2) - DRCANX(3)
          ANG23Y = DRCANY(2) - DRCANY(3)
          ANG23Z = DRCANZ(2) - DRCANZ(3)
          ANG23  = ANG23X + ANG23Y + ANG23Z

*          ANG12X = DRCANX(1) - DRCANX(2)
*          ANG12Y = DRCANY(1) - DRCANY(2)
*          ANG12Z = DRCANZ(1) - DRCANZ(2)

*     DOUBLE PENDULUM CALCULATION
          PA     = L3*M3P*SIN(TH2-TH3)*TH3DOT**2
          PB     = SIN(TH2)*G*(M3P+M2P)
          PC     = L3*M3P*COS(TH2-TH3)
          PD     = L2*(M3P+M2P)
```

```
          PE      = L2*SIN(TH2-TH3)*TH2DOT**2
          PF      = SIN(TH3)*G
          PG      = COS(TH2-TH3)*L2
          PH      = L3
          TH3DDT = (PE-PF+((PG*PA+PG*PB)/PD))/(PH-(PG*PC/PD))
          TH2DDT = (-PA - PB - PC*TH3DDT)/PD
          TH3DOT = INTGRL(TH3D0,TH3DDT)
          TH2DOT = INTGRL(TH2D0,TH2DDT)
          TH3     = INTGRL(TH30,TH3DOT)
          TH2     = INTGRL(TH20,TH2DOT)
          TH2ANG = TH2 * RADEG
          IF(TH2ANG.GT.0.0)THEN
           ANGTH2 = 180.0 - DRCANZ(2)
           ELSE
           ANGTH2 = DRCANZ(2) - 180.0
          ENDIF
          TH3ANG = TH3 * RADEG
          IF(TH3ANG.GT.0.0)THEN
           ANGTH3 = 180.0 - DRCANZ(3)
           ELSE
           ANGTH3 = DRCANZ(3) - 180.0
          ENDIF

END
STOP

FORTRAN

*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS

          SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
940           IMPLICIT REAL*8 (A-Z)
              DIMENSION VECTA(3),VECTB(3)
              MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
              MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
              MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
           RETURN

           END
```

APPENDIX C

THREE DIMENSION VALIDATION PROGRAM

```
TITLE NONSINGULAR 3-D VALIDATION PROGRAM

**********************************************************************************
*  THIS IS A PROGRAM WHICH VALIDATES THE NON-SINGULARITY OF A THREE      *
*  LINK RIGID BODY MANIPULATOR WITHOUT THE USE OF THE STANDARD ROBOT     *
*  TRANSFORMATION MATRICES.  THE ORIGINAL PROGRAM WAS WRITTEN BY LT      *
*  ATINOK AND HAS BEEN GREATLY MODIFIED TO INCLUDE THREE DIMENSIONAL     *
*  MOTION AND GRAVITY BY   LT  ROBERT M. VERBOS.        SEPTEMBER 1988  *
**********************************************************************************

TERMINAL
METHOD ADAMS
PRINT .02,ERROR1,ERROR2,ERROR3, ANG12, ANG23
CONTROL FINTIM =1.6, DELT = .0005, DELMAX = .1, DELPRT = .02
D     DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D     DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D     DIMENSION IMAT(3,3,3),NIMAT(3,3),TMAT(3,3),TMATTR(3,3),MATTMP(3,3)
D     DIMENSION LIMAT(3,3,3)
FIXED   IER,I,J,M,K,P,N,IA,IDGT,A,I1
ARRAY MATB(27),MATC(27),MATD(27),LCOGX(3),LCOGY(3),LCOGZ(3)
ARRAY VECTA(3),VECTB(3)
ARRAY WDX(3),WDY(3),WDZ(3),W1(3),W2(3),W3(3)
ARRAY RBG1(3),RAG1(3),RBG2(3),RAG2(3),RBG3(3)
ARRAY WKAREA(850)
ARRAY IXXT(3),IYYT(3),IZZT(3),IXYT(3),IXZT(3),IYZT(3)
ARRAY DRCANX(3),DRCANY(3),DRCANZ(3)
ARRAY DRCSX(3),DRCSY(3),DRCSZ(3),

INITIAL
******   INPUT   ******
*       INPUT PARAMETER CONSTANTS
55          A    =  15.0D0
            P    =  0.0D0
            W    =  PI/2.0
            IDGT =  6
            G    =  9.81D0
            N    = 27
            M    = 1
            IA   = 27
            IER  = 0
            LEVELQ = 0
            LEVLDQ = 0
*       INPUT JOINT LOCATIONS IN METERS
          JX0 = 0.0D0
          JY0 = 0.0D0
```

```
          JZ0 = 0.0D0
          JX1 = 0.0D0
          JY1 = 0.0D0
          JZ1 = 0.2D0
          JX2 = 0.0D0
          JY2 = 0.416D0
          JZ2 = 0.2D0
*       INPUT DISTANCE FROM CENTER OF LINK TO CENTER OF MASS
          L(1,1) = 0.05D0
          L(1,2) = 0.15D0
          L(2,1) = 0.208D0
          L(2,2) = 0.208D0
          L(3,1) = 0.2255D0
          L(3,2) = 0.2255D0
*       INPUT THE LINK LENGTHS
          LINKL1 = 0.20D0
          LINKL2 = 0.416D0
          LINKL3 = 0.451D0
*       INPUT MASS AT LINK ENDS IN KILOGRAMS
83        MASS(1,1) = 1.0D0
          MASS(1,2) = 3.0D0
          MASS(2,1) = 2.2D0
          MASS(2,2) = 2.2D0
          MASS(3,1) = 8.6D0
          MASS(3,2) = 8.6D0
*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
90          LCOGX(1) =  0.0D0
            X1       =  LCOGX(1)
            LCOGY(1) =  0.0D0
            Y1       =  LCOGY(1)
            LCOGZ(1) =  0.10D0
            Z1       =  LCOGZ(1)
            LCOGX(2) =  0.0D0
            X2       =  LCOGX(2)
            LCOGY(2) =  0.208D0
            Y2       =  LCOGY(2)
            LCOGZ(2) =  0.20D0
            Z2       =  LCOGZ(2)
            LCOGX(3) =  0.0D0
            X3       =  LCOGX(3)
            LCOGY(3) =  0.6415D0
            Y3       =  LCOGY(3)
            LCOGZ(3) =  0.2D0
            Z3       =  LCOGZ(3)
*       INPUT MASS OF EACH LINK IN KG
            M1 = 4.0D0
            M2 = 4.4D0
            M3 = 17.2D0
*       INPUT ACCELERATIONS OF JOINT ZERO
            AX0 = 0.0D0
            AY0 = 0.0D0
            AZ0 = 0.0D0
*       INPUT THE INITIAL DIRECTION COSINES
            DRCSX(1) = 0.0D0
            DRCSY(1) = 0.0D0
            DRCSZ(1) = 1.0D0
```

```
            DRCSX(2) = 0.0D0
            DRCSY(2) = 1.0D0
            DRCSZ(2) = 0.0D0
            DRCSX(3) = 0.0D0
            DRCSY(3) = 1.0D0
            DRCSZ(3) = 0.0D0
*       INPUT  THE INITIAL DIRECTION COSINE ANGLES
            DRCANX(1) = 90.0D0
            DRCANY(1) = 90.0D0
            DRCANZ(1) =  0.0D0
            DRCANX(2) = 90.0D0
            DRCANY(2) =  0.0D0
            DRCANZ(2) = 90.0D0
            DRCANX(3) = 90.0D0
            DRCANY(3) =  0.0D0
            DRCANZ(3) = 90.0D0

******    INITIALIZE   ******
*       OMEGA AND OMEGA DOT
139         DO 146 I = 1,3
               W1(I)  = 0.0D0
               W2(I)  = 0.0D0
               W3(I)  = 0.0D0
               WDX(I) = 0.0D0
               WDY(I) = 0.0D0
               WDZ(I) = 0.0D0
146         CONTINUE
               W1(3)  = 2.0D0
               W2(1)  = 2.0D0
               W3(1)  = 2.0D0
               WDY(2) = 4.0D0
               WDY(3) = 4.0D0

               THZ    = 0.0D0
*      INITIALIZE MATRICES TO ZERO
            DO 180 I = 1,27
               DO 175 J = 1,27
                  MATA(I,J) = 0.0D0
175            CONTINUE
               MATB(I) = 0.0D0
               MATC(I) = 0.0D0
               MATD(I) = 0.0D0
180         CONTINUE

*******     CALCULATIONS     ******
*       WEIGHTS (NEWTONS)
185         WG1 = M1*G
            WG2 = M2*G
            WG3 = M3*G
*       COMPUTE THE LENGTH FROM THE INBOARD JOINT TO COG
            LNCOG1 = DSQRT ( LCOGX(1)*LCOGX(1) + LCOGY(1)*LCOGY(1) +...
                             LCOGZ(1)*LCOGZ(1) )
            LX2 = LCOGX(2) - JX1
            LY2 = LCOGY(2) - JY1
            LZ2 = LCOGZ(2) - JZ1
            LNCOG2 = DSQRT ( LX2*LX2 + LY2*LY2 + LZ2*LZ2 )
```

```
          LX3 = LCOGX(3) - JX2
          LY3 = LCOGY(3) - JY2
          LZ3 = LCOGZ(3) - JZ2
          LNCOG3 = DSQRT ( LX3*LX3 + LY3*LY3 + LZ3*LZ3 )
*      COMPUTE FBD INERTIAS BASED ON POINT MASSES IN GLOBAL COORDINATES
190       DO 225 I = 1,3
             RX(I,1) = -L(I,1) * DRCSX(I)
             RX(I,2) =  L(I,2) * DRCSX(I)
             RY(I,1) = -L(I,1) * DRCSY(I)
             RY(I,2) =  L(I,2) * DRCSY(I)
             RZ(I,1) = -L(I,1) * DRCSZ(I)
             RZ(I,2) =  L(I,2) * DRCSZ(I)
200   IXX(I,1) = MASS(I,1) * ((RY(I,1) * RY(I,1)) + (RZ(I,1) * RZ(I,1)))
      IXX(I,2) = MASS(I,2) * ((RY(I,2) * RY(I,2)) + (RZ(I,2) * RZ(I,2)))
      IXXT(I)  = IXX(I,1) + IXX(I,2)
      IYY(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RZ(I,1) * RZ(I,1)))
      IYY(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RZ(I,2) * RZ(I,2)))
      IYYT(I)  = IYY(I,1) + IYY(I,2)
      IZZ(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RY(I,1) * RY(I,1)))
      IZZ(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RY(I,2) * RY(I,2)))
      IZZT(I)  = IZZ(I,1) + IZZ(I,2)
205   IXY(I,1) = MASS(I,1) * RX(I,1) * RY(I,1)
      IXY(I,2) = MASS(I,2) * RX(I,2) * RY(I,2)
      IXYT(I)  = IXY(I,1) + IXY(I,2)
      IXZ(I,1) = MASS(I,1) * RZ(I,1) * RX(I,1)
      IXZ(I,2) = MASS(I,2) * RZ(I,2) * RX(I,2)
      IXZT(I)  = IXZ(I,1) + IXZ(I,2)
      IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
      IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
      IYZT(I)  = IYZ(I,1) + IYZ(I,2)
      IF (IXXT(I) .LE. .01) THEN
      IXXT(I)  = .01
      ELSE
      IXXT(I) = IXXT(I)
      END IF
      IF (IYYT(I) .LE. .01) THEN
      IYYT(I)  = .01
      ELSE
      IYYT(I)  = IYYT(I)
      END IF
      IF (IZZT(I) .LE. .01) THEN
      IZZT(I)  = .01
      ELSE
      IZZT(I)  = IZZT(I)
      END IF
          IMAT(I,1,1) =  IXXT(I)
          IMAT(I,1,2) =  IXYT(I)
          IMAT(I,1,3) =  IXZT(I)
          IMAT(I,2,1) = -IXYT(I)
          IMAT(I,2,2) =  IYYT(I)
          IMAT(I,2,3) =  IYZT(I)
          IMAT(I,3,1) = -IXZT(I)
          IMAT(I,3,2) = -IYZT(I)
          IMAT(I,3,3) =  IZZT(I)
225       CONTINUE
*      DUE TO LINK 1 CONSTRAINTS AND SYMMETRY LINK 1 INERTIAS ARE CONST
```

```
          IXXT(1) = IMAT(1,1,1)
          IXYT(1) = IMAT(1,1,2)
          IXZT(1) = IMAT(1,1,3)
          IYYT(1) = IMAT(1,2,2)
          IYZT(1) = IMAT(1,2,3)
          IZZT(1) = IMAT(1,3,3)
*     TRANSFORM THE GLOBAL INERTIAS TO LOCAL  WHICH REMAIN CONSTANT
          DO 281 I = 2, 3
            TMAT(2,1) = -DCOS ( THZ )
            TMAT(2,2) = -DSIN ( THZ )
            TMAT(2,3) = 0.0D0
            TMAT(3,1) = DRCSX(I)
            TMAT(3,2) = DRCSY(I)
            TMAT(3,3) = DRCSZ(I)
            VECTA(1)  = TMAT(2,1)
            VECTA(2)  = TMAT(2,2)
            VECTA(3)  = TMAT(2,3)
            VECTB(1)  = TMAT(3,1)
            VECTB(2)  = TMAT(3,2)
            VECTB(3)  = TMAT(3,3)
            CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
            TMAT(1,1) = MI1
            TMAT(1,2) = MJ1
            TMAT(1,3) = MK1
            TMATTR(1,1) = TMAT(1,1)
            TMATTR(1,2) = TMAT(2,1)
            TMATTR(1,3) = TMAT(3,1)
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)
            DO 272 I1 = 1,3
            DO 272 J  = 1,3
               TEMP = 0.0D0
              DO 270  K  = 1,3
                 TEMP = TMAT(I1,K) * IMAT(I,K,J) + TEMP
270           CONTINUE
               MATTMP(I1,J) = TEMP
272         CONTINUE
            DO 280   I1 = 1,3
            DO 280   J  = 1,3
               TEMP = 0.0D0
              DO 278   K  = 1,3
                 TEMP = MATTMP(I1,K) * TMATTR(K,J) + TEMP
278           CONTINUE
             LIMAT(I,I1,J) = TEMP
280         CONTINUE
281       CONTINUE

DERIVATIVE
NOSORT
287         CALL ERRSET (208,256,-1,1,1)
            CALL UERSET(LEVELQ,LEVLDQ)
*      INITIALIZE MATRICES TO ZERO
```

```
          DO 297 I = 1,27
             DO 293 J = 1,27
                MATA(I,J) = 0.0D0
293           CONTINUE
              MATB(I) = 0.0D0
              MATC(I) = 0.0D0
              MATD(I) = 0.0D0
297        CONTINUE

*    VALIDATION PORTION OF PROGRAM  ***
*  ENTER THE ANGULAR AND LINEAR ACCELERATIONS, AND FORCES INTO MATRIX C
*  FOR THE CONSTRAIN OF THE NEPTUNE II
         TH1L     =  2.0 * TIME + PI/2.0
         W1L      =  2.0D0
         WD1L     =  0.0D0
         TH1ANG   =  TH1L * RADEG
         WDX1G    =  0.0D0
         WDY1G    =  0.0D0
         WDZ1G    =  WD1L
         WX1G     =  0.0D0
         WY1G     =  0.0D0
         WZ1G     =  W1L
         TH2L     =  2.0 * TIME
         W2L      =  2.0
         WD2L     =  0.0D0
         TH2ANG   =  TH2L * RADEG
         WX2G     =  W2L * SIN(TH1L)
         WY2G     =  W2L * (- COS(TH1L) )
         WZ2G     =  W1L
         VECTA(1) =  0.0D0
         VECTA(2) =  0.0D0
         VECTA(3) =  WZ2G
         VECTB(1) =  WX2G
         VECTB(2) =  WY2G
         VECTB(3) =  0.0D0
         CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
         WDX2G    =  MI1
         WDY2G    =  MJ1
         WDZ2G    =  MK1
         TH3L     =  0.0D0
         W3L      =  0.0D0
         WD3L     =  0.0D0
         TH3ANG   =  TH3L * RADEG
         WX3G     =  WX2G
         WY3G     =  WY2G
         WZ3G     =  WZ2G
         WDX3G    =  WDX2G
         WDY3G    =  WDY2G
         WDZ3G    =  WDZ2G
         R2X      =  LNCOG2 * COS(TH2L) * COS(TH1L)
         R2Y      =  LNCOG2 * COS(TH2L) * SIN(TH1L)
         R2Z      =  LNCOG2 * SIN(TH2L)
         RJ2X     =  LINKL2 * COS(TH2L) * COS(TH1L)
         RJ2Y     =  LINKL2 * COS(TH2L) * SIN(TH1L)
         RJ2Z     =  LINKL2 * SIN(TH2L)
         R3X      =  RJ2X + LNCOG3 * COS(TH2L+TH3L) * COS(TH1L)
```

```
      R3Y      =  RJ2Y + LNCOG3 * COS(TH2L+TH3L) * SIN(TH1L)
      R3Z      =  RJ2Z + LNCOG3 * SIN(TH2L+TH3L)
      LTIPX    =  RJ2X + LINKL3 * COS(TH2L+TH3L) * COS(TH1L)
      LTIPY    =  RJ2Y + LINKL3 * COS(TH2L+TH3L) * SIN(TH1L)
      LTIPZ    =  RJ2Z + LINKL3 * SIN(TH2L+TH3L) + JZ1
*     CALCULATE THE GLOBAL ACCELERATIONS
      AX1G     =  0.0D0
      AY1G     =  0.0D0
      AZ1G     =  0.0D0
      VECTA(1) =  WDX2G
      VECTA(2) =  WDY2G
      VECTA(3) =  WDZ2G
      VECTB(1) =  R2X
      VECTB(2) =  R2Y
      VECTB(3) =  R2Z
      CALL CPROD  (VECTA,VECTB,MI1,MJ1,MK1)
      VECTA(1) =  WX2G
      VECTA(2) =  WY2G
      VECTA(3) =  WZ2G
      CALL CPROD  (VECTA,VECTB,MI2,MJ2,MK2)
      VECTB(1) =  MI2
      VECTB(2) =  MJ2
      VECTB(3) =  MK2
      CALL CPROD  (VECTA,VECTB,MI2,MJ2,MK2)
      AX2G     = MI1 + MI2
      AY2G     = MJ1 + MJ2
      AZ2G     = MK1 + MK2
      VECTA(1) =  WDX3G
      VECTA(2) =  WDY3G
      VECTA(3) =  WDZ3G
      VECTB(1) =  R3X
      VECTB(2) =  R3Y
      VECTB(3) =  R3Z
      CALL CPROD  (VECTA,VECTB,MI1,MJ1,MK1)
      VECTA(1) =  WX3G
      VECTA(2) =  WY3G
      VECTA(3) =  WZ3G
      CALL CPROD  (VECTA,VECTB,MI2,MJ2,MK2)
      VECTB(1) =  MI2
      VECTB(2) =  MJ2
      VECTB(3) =  MK2
      CALL CPROD  (VECTA,VECTB,MI2,MJ2,MK2)
      AX3G     = MI1 + MI2
      AY3G     = MJ1 + MJ2
      AZ3G     = MK1 + MK2
*     SET MATRIX D
      MATD(27)  =  WDZ3G
      MATD(26)  =  WDY3G
      MATD(25)  =  WDX3G
      MATD(24)  =  AZ3G
      MATD(23)  =  AY3G
      MATD(22)  =  AX3G
      MATD(21)  = -M3 * AZ3G - WG3
      MATD(20)  = -M3 * AY3G
      MATD(19)  = -M3 * AX3G
      MATD(18)  =  WDZ2G
```

```
          MATD(17)  =  WDY2G
          MATD(16)  =  WDX2G
          MATD(15)  =  AZ2G
          MATD(14)  =  AY2G
          MATD(13)  =  AX2G
          MATD(12)  =  MATD(21) - M2 * AZ2G - WG2
          MATD(11)  =  MATD(20) - M2 * AY2G
          MATD(10)  =  MATD(19) - M2 * AX2G
          MATD(9)   =  WDZ1G
          MATD(8)   =  WDY1G
          MATD(7)   =  WDX1G
          MATD(6)   =  AZ1G
          MATD(5)   =  AY1G
          MATD(4)   =  AX1G
          MATD(3)   =  MATD(12) - M1 * AZ1G  - WG1
          MATD(2)   =  MATD(11) - M1 * AY1G
          MATD(1)   =  MATD(10) - M1 * AX1G

****  DIRECT DYNAMICS PORTION OF PROGRAM  ****
*       INPUT JOINT EQUATIONS
*       JOINT ZERO  EQUATIONS
*        W1 X (W1 X RB/G1)
                RBG1(1)  = JX0 - LCOGX(1)
                RBG1(2)  = JY0 - LCOGY(1)
                RBG1(3)  = JZ0 - LCOGZ(1)
                VECTA(1) = W1(1)
                VECTA(2) = W1(2)
                VECTA(3) = W1(3)
             CALL CPROD(VECTA,RBG1,MIC0,MJC0,MKC0)
                VECTB(1) = MIC0
                VECTB(2) = MJC0
                VECTB(3) = MKC0
             CALL CPROD(VECTA,VECTB,MIC0,MJC0,MKC0)
*       W1 X (W1 X RA/G1)
                RAG1(1) = JX1 - LCOGX(1)
                RAG1(2) = JY1 - LCOGY(1)
                RAG1(3) = JZ1 - LCOGZ(1)
                VECTA(1) = W1(1)
                VECTA(2) = W1(2)
                VECTA(3) = W1(3)
             CALL CPROD (VECTA,RAG1,MIC1,MJC1,MKC1)
                VECTB(1) = MIC1
                VECTB(2) = MJC1
                VECTB(3) = MKC1
             CALL CPROD (VECTA,VECTB,MIC1,MJC1,MKC1)
*       W2 X (W2 X RB/G2)
                RBG2(1) = JX1 - LCOGX(2)
                RBG2(2) = JY1 - LCOGY(2)
                RBG2(3) = JZ1 - LCOGZ(2)
                VECTA(1) = W2(1)
                VECTA(2) = W2(2)
                VECTA(3) = W2(3)
             CALL CPROD (VECTA,RBG2,MIC2,MJC2,MKC2)
                VECTB(1) = MIC2
                VECTB(2) = MJC2
                VECTB(3) = MKC2
```

```
            CALL CPROD (VECTA,VECTB,MIC2,MJC2,MKC2)
*       W2 X (W2 X RA/G2)
               RAG2(1) = JX2 - LCOGX(2)
               RAG2(2) = JY2 - LCOGY(2)
               RAG2(3) = JZ2 - LCOGZ(2)
               VECTA(1) = W2(1)
               VECTA(2) = W2(2)
               VECTA(3) = W2(3)
            CALL CPROD (VECTA,RAG2,MIC3,MJC3,MKC3)
               VECTB(1) = MIC3
               VECTB(2) = MJC3
               VECTB(3) = MKC3
            CALL CPROD(VECTA,VECTB,MIC3,MJC3,MKC3)
*       W3 X (W3 X RB/G3)
               RBG3(1) = JX2 - LCOGX(3)
               RBG3(2) = JY2 - LCOGY(3)
               RBG3(3) = JZ2 - LCOGZ(3)
               VECTA(1) = W3(1)
               VECTA(2) = W3(2)
               VECTA(3) = W3(3)
            CALL CPROD (VECTA,RBG3,MIC4,MJC4,MKC4)
               VECTB(1) = MIC4
               VECTB(2) = MJC4
               VECTB(3) = MKC4
            CALL CPROD (VECTA,VECTB,MIC4,MJC4,MKC4)
*       INERTIA TRANSFORMATION
            TMAT(2,1) = -DCOS ( THZ )
            TMAT(2,2) = -DSIN ( THZ )
            TMAT(2,3) = 0.0D0
          DO 880 I = 2, 3
            TMAT(3,1) = DRCSX(I)
            TMAT(3,2) = DRCSY(I)
            TMAT(3,3) = DRCSZ(I)
            VECTA(1)  = TMAT(2,1)
            VECTA(2)  = TMAT(2,2)
            VECTA(3)  = TMAT(2,3)
            VECTB(1)  = TMAT(3,1)
            VECTB(2)  = TMAT(3,2)
            VECTB(3)  = TMAT(3,3)
            CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
            TMAT(1,1) = MI1
            TMAT(1,2) = MJ1
            TMAT(1,3) = MK1
            TMATTR(1,1) = TMAT(1,1)
            TMATTR(1,2) = TMAT(2,1)
            TMATTR(1,3) = TMAT(3,1)
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)
            DO 866 I1 = 1,3
            DO 866 J  = 1,3
               TEMP = 0.0D0
              DO 864  K  = 1,3
```

```
                TEMP = TMATTR(I1,K) * LIMAT(I,K,J) + TEMP
864           CONTINUE
               MATTMP(I1,J) = TEMP
866         CONTINUE
            DO 874 I1 = 1,3
            DO 874 J  = 1,3
               TEMP = 0.0D0
              DO 872 K  = 1,3
                 TEMP = MATTMP(I1,K) * TMAT(K,J) + TEMP
872           CONTINUE
             NIMAT(I1,J) = TEMP
874         CONTINUE
             IXXT(I) = NIMAT(1,1)
             IXYT(I) = NIMAT(1,2)
             IXZT(I) = NIMAT(1,3)
             IYYT(I) = NIMAT(2,2)
             IYZT(I) = NIMAT(2,3)
             IZZT(I) = NIMAT(3,3)
880     CONTINUE

*    ENTER CONSTANS INTO MATRIX A
**       LINK ONE
*        SUM OF FORCES
*        X DIRECTION
887        MATA(1,1)  =  1.0D0
           MATA(1,4)  =  M1
           MATA(1,10) = -1.0D0
*        Y DIRECTION
           MATA(2,2)  =  1.0D0
           MATA(2,5)  =  M1
           MATA(2,11) = -1.0D0
*        Z DIRECTION
           MATA(3,3)  =  1.0D0
           MATA(3,6)  =  M1
           MATA(3,12) = -1.0D0
*        EQUATIONS AT JOINT ZERO
*        X DIRECTION
           MATA(4,4) =  1.0D0
           MATA(4,8) =  RBG1(3)
           MATA(4,9) = -RBG1(2)
*        Y DIRECTION
           MATA(5,5) =  1.0D0
           MATA(5,7) = -RBG1(3)
           MATA(5,9) =  RBG1(1)
*        Z DIRECTION
           MATA(6,6) =  1.0D0
           MATA(6,7) =  RBG1(2)
           MATA(6,8) = -RBG1(1)
*        SUM OF MOMENTS EQUATIONS
*        X DIRECTION
           MATA(7,2)  =  RBG1(3)
           MATA(7,3)  = -RBG1(2)
           MATA(7,7)  = -IXXT(1)
           MATA(7,8)  =  IXYT(1)
           MATA(7,9)  =  IXZT(1)
           MATA(7,11) = -RAG1(3)
```

```
          MATA(7,12) =  RAG1(2)
*       Y DIRECTION
          MATA(8,1)  = -RBG1(3)
          MATA(8,3)  =  RBG1(1)
          MATA(8,7)  =  IXYT(1)
          MATA(8,8)  = -IYYT(1)
          MATA(8,9)  =  IYZT(1)
          MATA(8,10) =  RAG1(3)
          MATA(8,12) = -RAG1(1)
*       Z DIRECTION
          MATA(9,1)  =  RBG1(2)
          MATA(9,2)  = -RBG1(1)
          MATA(9,7)  =  IXZT(1)
          MATA(9,8)  =  IYZT(1)
          MATA(9,9)  = -IZZT(1)
          MATA(9,10) = -RAG1(2)
          MATA(9,11) =  RAG1(1)
**      LINK TWO
*       SUM OF FORCES
*       X DIRECTION
939       MATA(10,10) =  1.0D0
          MATA(10,13) =  M2
          MATA(10,19) = -1.0D0
*       Y DIRECTION
          MATA(11,11) =  1.0D0
          MATA(11,14) =  M2
          MATA(11,20) = -1.0D0
*       Z DIRECTION
          MATA(12,12) =  1.0D0
          MATA(12,15) =  M2
          MATA(12,21) = -1.0D0
*       EQUATIONS AT JOINT ONE
*       X DIRECTION
          MATA(13,4)  = -1.0D0
          MATA(13,8)  = -RAG1(3)
          MATA(13,9)  =  RAG1(2)
          MATA(13,13) =  1.0D0
          MATA(13,17) =  RBG2(3)
          MATA(13,18) = -RBG2(2)
*       Y DIRECTION
          MATA(14,5)  = -1.0D0
          MATA(14,7)  =  RAG1(3)
          MATA(14,9)  = -RAG1(1)
          MATA(14,14) =  1.0D0
          MATA(14,16) = -RBG2(3)
          MATA(14,18) =  RBG2(1)
*       Z DIRECTION
          MATA(15,6)  = -1.0D0
          MATA(15,7)  = -RAG1(2)
          MATA(15,8)  =  RAG1(1)
          MATA(15,15) =  1.0D0
          MATA(15,16) =  RBG2(2)
          MATA(15,17) = -RBG2(1)
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(16,11) =  RBG2(3)
```

```
          MATA(16,12) = -RBG2(2)
          MATA(16,16) = -IXXT(2)
          MATA(16,17) =  IXYT(2)
          MATA(16,18) =  IXZT(2)
          MATA(16,20) = -RAG2(3)
          MATA(16,21) =  RAG2(2)
*       Y DIRECTION
          MATA(17,10) = -RBG2(3)
          MATA(17,12) =  RBG2(1)
          MATA(17,16) =  IXYT(2)
          MATA(17,17) = -IYYT(2)
          MATA(17,18) =  IYZT(2)
          MATA(17,19) =  RAG2(3)
          MATA(17,21) = -RAG2(1)
*       Z DIRECTION
          MATA(18,10) =  RBG2(2)
          MATA(18,11) = -RBG2(1)
          MATA(18,16) =  IXZT(2)
          MATA(18,17) =  IYZT(2)
          MATA(18,18) = -IZZT(2)
          MATA(18,19) = -RAG2(2)
          MATA(18,20) =  RAG2(1)
**      LINK THREE
*       SUM OF FORCES
*       X DIRECTION
1000      MATA(19,19) =  1.0D0
          MATA(19,22) =  M3
*       Y DIRECTION
          MATA(20,20) =  1.0D0
          MATA(20,23) =  M3
*       Z DIRECTION
          MATA(21,21) =  1.0D0
          MATA(21,24) =  M3
*       EQUATIONS AT JOINT TWO
*       X DIRECTION
          MATA(22,13) = -1.0D0
          MATA(22,17) = -RAG2(3)
          MATA(22,18) =  RAG2(2)
          MATA(22,22) =  1.0D0
          MATA(22,26) =  RBG3(3)
          MATA(22,27) = -RBG3(2)
*       Y DIRECTION
          MATA(23,14) = -1.0D0
          MATA(23,16) =  RAG2(3)
          MATA(23,18) = -RAG2(1)
          MATA(23,23) =  1.0D0
          MATA(23,25) = -RBG3(3)
          MATA(23,27) =  RBG3(1)
*       Z DIRECTION
          MATA(24,15) = -1.0D0
          MATA(24,16) = -RAG2(2)
          MATA(24,17) =  RAG2(1)
          MATA(24,24) =  1.0D0
          MATA(24,25) =  RBG3(2)
```

```
          MATA(24,26) = -RBG3(1)
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(25,20) =  RBG3(3)
          MATA(25,21) = -RBG3(2)
          MATA(25,25) = -IXXT(3)
          MATA(25,26) =  IXYT(3)
          MATA(25,27) =  IXZT(3)
*       Y DIRECTION
          MATA(26,19) = -RBG3(3)
          MATA(26,21) =  RBG3(1)
          MATA(26,25) =  IXYT(3)
          MATA(26,26) = -IYYT(3)
          MATA(26,27) =  IYZT(3)
*       Z DIRECTION
          MATA(27,19) =  RBG3(2)
          MATA(27,20) = -RBG3(1)
          MATA(27,25) =  IXZT(3)
          MATA(27,26) =  IYZT(3)
          MATA(27,27) = -IZZT(3)
******   FIRST LINK IS CONSTRAINED TO ROTATE IN Z DIRECTION   ******
1051      DO 1056 I = 4,8
            DO 1055 J = 1,27
              MATA(I,J) = 0.0D0
*             MATA(J,I) = 0.0D0
1055        CONTINUE
1056      CONTINUE
              MATA(4,4) = 1.0D0
              MATA(5,5) = 1.0D0
              MATA(6,6) = 1.0D0
              MATA(7,7) = 1.0D0
              MATA(8,8) = 1.0D0
          DO 1065 J = 1,27
             MATA(18,J) = 0.0D0
             MATA(27,J) = 0.0D0
1065      CONTINUE
            IZZT(2) = IZZT(2) + M2 * (DSQRT(LCOGX(2)**2+LCOGY(2)**2))**2
            IZZT(3) = IZZT(3) + M3 * (DSQRT(LCOGX(3)**2+LCOGY(3)**2))**2
            MATA(9,9)   = -(IZZT(1)+IZZT(2)+IZZT(3))
            MATA(18,9)  = -1.0D0
            MATA(18,18) =  1.0D0
            MATA(27,9)  = -1.0D0
            MATA(27,27) =  1.0D0
*  MULTIPLY MATA * MATD TO OBTAIN TORQUES
          DO 758 J = 1,27
               SUM1 = 0.0D0
               DO 755 K = 1,27
                    SUM1 = SUM1 + MATA(J,K) * MATD(K)
755            CONTINUE
               MATC(J) = SUM1
758       CONTINUE
          T2Z  =  -MATC(27)
          T2Y  =  -MATC(26)
          T2X  =  -MATC(25)
          T1Z  =   T2Z - MATC(18)
          T1Y  =   T2Y - MATC(17)
```

```
          T1X  =   T2X - MATC(16)
          T0Z  =   T1Z - MATC(9)
          T0Y  =   T1Y - MATC(8)
          T0X  =   T1X - MATC(7)
*     ENTER KNOWNS INTO MATB
1077      MATB(1)    =  0.0D0
          MATB(2)    =  0.0D0
          MATB(3)    = -WG1
          MATB(4)    =  AX0 - MIC0
          MATB(5)    =  AY0 - MJC0
          MATB(6)    =  AZ0 - MKC0
          MATB(7)    =  T1X - T0X
          MATB(8)    =  T1Y - T0Y
          MATB(9)    =  T1Z - T0Z
          MATB(10)   =  0.0D0
          MATB(11)   =  0.0D0
          MATB(12)   = -WG2
          MATB(13)   =  MIC1 - MIC2
          MATB(14)   =  MJC1 - MJC2
          MATB(15)   =  MKC1 - MKC2
          MATB(16)   =  T2X - T1X
          MATB(17)   =  T2Y - T1Y
          MATB(18)   =  T2Z - T1Z
          MATB(19)   =  0.0D0
          MATB(20)   =  0.0D0
          MATB(21)   = -WG3
          MATB(22)   =  MIC3 - MIC4
          MATB(23)   =  MJC3 - MJC4
          MATB(24)   =  MKC3 - MKC4
          MATB(25)   = -T2X
          MATB(26)   = -T2Y
          MATB(27)   = -T2Z
******    FIRST LINK IS CONSTRAINED TO ROTATE IN Z DIRECTION   ******
          DO 1107 I = 4,8
            MATB(I)   =  0.0D0
1107      CONTINUE
            MATB(18)  =  0.0D0
            MATB(27)  =  0.0D0

*       CALL EQUATION SOLVER PROGRAM FROM IMSL
1112        CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
            IF (IER.EQ.0) THEN
            GOTO 1133
            ELSE
            WRITE (*,1117) IER
1117        FORMAT (I5)
            DO 1127  I = 1, 27
            WRITE (*,1120) I
1120        FORMAT (I7)
            DO 1124  J = 1, 27, 3
            WRITE (*,1123) J,MATA(I,J),J+1,MATA(I,J+1),J+2,MATA(I,J+2)
1123        FORMAT (I5,F13.5,I5,F13.5,I5,F13.5)
1124        CONTINUE
            WRITE (*,1126) I,MATB(I)
1126        FORMAT (I3,F15.5)
1127        CONTINUE
```

```
            CALL ENDJOB
            END IF
***     FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
*       JOINT ZERO
1133       FX0 = MATB(1)
           FY0 = MATB(2)
           FZ0 = MATB(3)
*       LINK ONE
*       SINCE LINK1 IS CONSTRAIN TO ROTATE AROUND THE Z ONLY
           AX1 = 0.0D0
           AY1 = 0.0D0
           AZ1 = 0.0D0
*141       AX1        = MATB(4)
*          VELX1      = INTGRL(0.,AX1)
*          LCOGX1     = INTGRL(X1,VELX1)
*          LCOGX(1)   = LCOGX1
*          AY1        = MATB(5)
*          VELY1      = INTGRL(0.,AY1)
*          LCOGY1     = INTGRL(Y1,VELY1)
*          LCOGY(1)   = LCOGY1
*          AZ1        = MATB(6)
*          VELZ1      = INTGRL(0.,AZ1)
*          LCOGZ1     = INTGRL(Z1,VELZ1)
*          LCOGZ(1)   = LCOGZ1
         WD1X       = MATB(7)
         W1X        = INTGRL(0.,WD1X)
         WDX(1)     = WD1X
         W1(1)      = W1X
         WD1Y       = MATB(8)
         W1Y        = INTGRL(0.,WD1Y)
         WDY(1)     = WD1Y
         W1(2)      = W1Y
         WD1Z       = MATB(9)
         W1Z        = INTGRL(2.,WD1Z)
         WDZ(1)     = WD1Z
         W1(3)      = W1Z
1166     THZ        = INTGRL(0.,W1Z)
         THZANG     = THZ * RADEG
         COSTHZ     = DCOS(THZ)
         SINTHZ     = DSIN(THZ)
*   IF THE 1ST LINK IS CONSTRAIN TO ROTATE IN THE Z DIRECTION ONLY
*   THE DIRECTION COSINE AND DIRECTION COSINE  ANGLES ARE CONSTANT
*   AND DO NOT NEED TO BE CALCULATED
*       JOINT ONE
1174     FX1 = MATB(10)
         FY1 = MATB(11)
         FZ1 = MATB(12)
**      LINK TWO
1178     AX2        = MATB(13)
         VELX2      = INTGRL(-0.416,AX2)
         LCOGX2     = INTGRL(X2,VELX2)
         LCOGX(2)   = LCOGX2
         AY2        = MATB(14)
         VELY2      = INTGRL(0.,AY2)
         LCOGY2     = INTGRL(Y2,VELY2)
         LCOGY(2)   = LCOGY2
```

```
          AZ2       = MATB(15)
          VELZ2     = INTGRL(0.416,AZ2)
          LCOGZ2    = INTGRL(Z2,VELZ2)
          LCOGZ(2)  = LCOGZ2
          WD2X      = MATB(16)
          W2X       = INTGRL(2.,WD2X)
          WDX(2)    = WD2X
          W2(1)     = W2X
          WD2Y      = MATB(17)
          W2Y       = INTGRL(0.,WD2Y)
          WDY(2)    = WD2Y
          W2(2)     = W2Y
          WD2Z      = MATB(18)
          W2Z       = INTGRL(2.,WD2Z)
          W2(3)     = W2Z
          WDZ(2)    = WD2Z
*       GET THE DIRECTION COSINES FOR THE LINK TWO
          DRCSX(2)  = (LCOGX2 - JX1) / LNCOG2
          DRCSY(2)  = (LCOGY2 - JY1) / LNCOG2
          DRCSZ(2)  = (LCOGZ2 - JZ1) / LNCOG2
          AMP = DSQRT(DRCSX(2)*DRCSX(2)+DRCSY(2)*DRCSY(2)+...
                      DRCSZ(2)*DRCSZ(2))
          DRCSX(2) = DRCSX(2)/AMP
          DRCSY(2) = DRCSY(2)/AMP
          DRCSZ(2) = DRCSZ(2)/AMP
          DRCANX(2) = DACOS(DRCSX(2)) * RADEG
          DRCANY(2) = DACOS(DRCSY(2)) * RADEG
          DRCANZ(2) = DACOS(DRCSZ(2)) * RADEG
*       JOINT LOCATION
1215      JX2 = JX1 + LINKL2 * DRCSX(2)
          JY2 = JY1 + LINKL2 * DRCSY(2)
          JZ2 = JZ1 + LINKL2 * DRCSZ(2)

*       JOINT TWO
1220      FX2 = MATB(19)
          FY2 = MATB(20)
          FZ2 = MATB(21)
**       LINK THREE
1224      AX3       = MATB(22)
          VELX3     = INTGRL(-1.282,AX3)
          LCOGX3    = INTGRL(X3,VELX3)
          LCOGX(3)  = LCOGX3
          AY3       = MATB(23)
          VELY3     = INTGRL(0.,AY3)
          LCOGY3    = INTGRL(Y3,VELY3)
          LCOGY(3)  = LCOGY3
          AZ3       = MATB(24)
          VELZ3     = INTGRL(1.282,AZ3)
          LCOGZ3    = INTGRL(Z3,VELZ3)
          LCOGZ(3)  = LCOGZ3
          WD3X      = MATB(25)
          W3X       = INTGRL(2.,WD3X)
          WDX(3)    = WD3X
          W3(1)     = W3X
          WD3Y      = MATB(26)
          W3Y       = INTGRL(0.,WD3Y)
```

```
          WDY(3)    = WD3Y
          W3(2)     = W3Y
          WD3Z      = MATB(27)
          W3Z       = INTGRL(2.,WD3Z)
          WDZ(3)    = WD3Z
          W3(3)     = W3Z
*  CALC DIRECTIONAL COSINES FOR LINK THREE
1249      DRCSX(3) = (LCOGX3 - JX2) / LNCOG3
          DRCSY(3) = (LCOGY3 - JY2) / LNCOG3
          DRCSZ(3) = (LCOGZ3 - JZ2) / LNCOG3
          AMP = DSQRT(DRCSX(3)*DRCSX(3)+DRCSY(3)*DRCSY(3)+...
                      DRCSZ(3)*DRCSZ(3))
          DRCSX(3) = DRCSX(3)/AMP
          DRCSY(3) = DRCSY(3)/AMP
          DRCSZ(3) = DRCSZ(3)/AMP
          DRCANX(3) = DACOS(DRCSX(3)) * RADEG
          DRCANY(3) = DACOS(DRCSY(3)) * RADEG
          DRCANZ(3) = DACOS(DRCSZ(3)) * RADEG
*   TIP LOCATION
1261      TIPX = JX2 + LINKL3 * DRCSX(3)
          TIPY = JY2 + LINKL3 * DRCSY(3)
          TIPZ = JZ2 + LINKL3 * DRCSZ(3)
*         FIND THE ANGLE BETWEEN THE LIMKS
1265      ANG23X = DRCANX(2) - DRCANX(3)
          ANG23Y = DRCANY(2) - DRCANY(3)
          ANG23Z = DRCANZ(2) - DRCANZ(3)
          ANG12X = DRCANX(1) - DRCANX(2)
          ANG12Y = DRCANY(1) - DRCANY(2)
          ANG12Z = DRCANZ(1) - DRCANZ(2)
          ANG12  = DRCANZ(2) - DRCANZ(1)
          ANG23  = ANG23X + ANG23Y + ANG23Z

*       CLACULATE THE ERROR BETWEEN KNOWN TIP POSITION AND CALCULATED
          ERROR1 = ABS ( LTIPX - TIPX ) / 0.867 * 100.0
          ERROR2 = ABS ( LTIPY - TIPY ) / 0.867 * 100.0
          ERROR3 = ABS ( LTIPZ - TIPZ ) / 0.867 * 100.0

END
STOP

FORTRAN
*     SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS

          SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
1284          IMPLICIT REAL*8 (A-Z)
              DIMENSION VECTA(3),VECTB(3)
              MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
              MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
              MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
           RETURN
           END
```

APPENDIX D

ROBOT VALIDATION PROGRAM

```
TITLE ROBOT TORQUE VALIDATION PROGRAM

**************************************************************************
*  THIS IS A PROGRAM THAT USES ACTUAL DATA FROM THE NEPTUNE II ROBOT TO *
*  OBTAIN TORQUES TO INPUT INTO THE NONSINGULAR SIMULATION OF A 3 LINK  *
*  RIGID MANIPULATOR.  THE ORIGINAL PROGRAM WAS WRITTEN BY LT ALTINOK   *
*  AND HAS BEEN GREATLY MODIFIED TO INCLUDE 3 D MOTION AND GRAVITATIONAL*
*  EFFECTS BY  LT  R. M.  VERBOS   USN        SEPTEMBER 1988            *
**************************************************************************

TERMINAL
METHOD RKSFX
PRINT .10,DRCANZ(3)
CONTROL FINTIM =7.0, DELT = .01, DELMAX = .1, DELPRT = .10
SAVE .10, DRCANZ(3)
GRAPH (DE=TEK618) TIME,DRCANZ(3)
D     DIMENSION MATA(27,27),MASS(3,2),L(3,2),RX(3,2),RY(3,2),RZ(3,2)
D     DIMENSION IXX(3,2),IXZ(3,2),IXY(3,2),IYY(3,2),IYZ(3,2),IZZ(3,2)
D     DIMENSION IMAT(3,3,3),NIMAT(3,3),TMAT(3,3),TMATTR(3,3),MATTMP(3,3)
D     DIMENSION LIMAT(3,3,3)
FIXED   IER,I,J,M,K,P,N,IA,IDGT,A,I1
ARRAY MATB(27),LCOGX(3),LCOGY(3),LCOGZ(3)
ARRAY VECTA0(3),VECTB0(3),VECTA1(3),VECTB1(3),VECTA2(3),VECTB2(3)
ARRAY VECTA(3),VECTB(3)
ARRAY WDX(3),WDY(3),WDZ(3),W1(3),W2(3),W3(3)
ARRAY RBG1(3),RAG1(3),RBG2(3),RAG2(3),RBG3(3)
ARRAY WKAREA(850)
ARRAY IXXT(3),IYYT(3),IZZT(3),IXYT(3),IXZT(3),IYZT(3)
ARRAY DRCANX(3),DRCANY(3),DRCANZ(3)
ARRAY DRCSX(3),DRCSY(3),DRCSZ(3)

INITIAL
******   INPUT   ******
*       INPUT PARAMETER CONSTANTS
          IDGT = 6
          G = 9.81D0
          N = 27
          M = 1
          IA = 27
          IER = 0
          LEVELQ = 0
          LEVLDQ = 0

*       INPUT DISTANCE FROM COG OF LINK TO CENTER OF MASS
*       FOR EACH LINK ENDS
```

```
          L(1,1) = 0.1953D0
          L(1,2) = 0.0651D0
          L(2,1) = 0.208D0
          L(2,2) = 0.208D0
          L(3,1) = 0.1746D0
*         L(3,2) = 0.1746D0

* INPUT THE LINK LENGTHS OF THE ROBOT
          LINKL1 = 0.2604D0
          LINKL2 = 0.416D0
          LINKL3 = 0.3498D0

*       INPUT JOINT LOCATIONS IN METERS
          JX0 = 0.0D0
          JY0 = 0.0D0
          JZ0 = 0.0D0
          JX1 = 0.0D0
          JY1 = 0.0D0
          JZ1 = 0.2604D0
          JX2 = 0.0D0
          JY2 = 0.416D0
          JZ2 = 0.2604D0

*       INPUT TORQUE CONSTANTS
          T0X    = 0.0D0
          T0Y    = 0.0D0
          T0Z    = 0.0D0
          T1X    = 0.0D0
          T1Y    = 0.0D0
          T1Z    = 0.0D0
          T2X    = 0.0D0
          T2Y    = 0.0D0
          T2Z    = 0.0D0
          TG1    = 0.0D0
          TG2    = 0.0D0
          T1FNC  = 0.0D0
          T2FNC  = 0.0D0

*       INPUT MASS AT LINK ENDS IN KILOGRAMS
110       MASS(1,1) = 2.273D0
          MASS(1,2) = 6.818D0
          MASS(2,1) = 0.455D0
          MASS(2,2) = 0.455D0
          MASS(3,1) = 5.909D0
          MASS(3,2) = 5.909D0

*       INPUT LOCATION OF LINK CENTERS OF GRAVITY
120        LCOGX(1) =  0.0D0
           X1       =  LCOGX(1)
           LCOGY(1) =  0.0D0
           Y1       =  LCOGY(1)
           LCOGZ(1) =  0.10D0
           Z1       =  LCOGZ(1)
           LCOGX(2) =  0.0D0
           X2       =  LCOGX(2)
           LCOGY(2) =  0.208D0
```

```
            Y2       =  LCOGY(2)
            LCOGZ(2) =  0.2604D0
            Z2       =  LCOGZ(2)
            LCOGX(3) =  0.0D0
            X3       =  LCOGX(3)
            LCOGY(3) =  0.4159D0
            Y3       =  LCOGY(3)
            LCOGZ(3) =  0.0858D0
            Z3       =  LCOGZ(3)

*       INPUT MASS OF EACH LINK IN KG
            M1 = 9.091D0
            M2 = 0.910D0
            M3 = 11.818D0

*       INPUT ACCELERATIONS OF JOINT ZERO
            AX0 = 0.0D0
            AY0 = 0.0D0
            AZ0 = 0.0D0

*       INPUT THE INITIAL DIRECTION COSINES
            DRCSX(1) = 0.0D0
            DRCSY(1) = 0.0D0
            DRCSZ(1) = 1.0D0
            DRCSX(2) = 0.0D0
            DRCSY(2) = 1.0D0
            DRCSZ(2) = 0.0D0
            DRCSX(3) = 0.0D0
            DRCSY(3) = 0.0D0
            DRCSZ(3) = -1.0D0

*       INPUT  THE INITIAL DIRECTION COSINE ANGLES
            DRCANX(1) = 90.0D0
            DRCANY(1) = 90.0D0
            DRCANZ(1) =  0.0D0
            DRCANX(2) = 90.0D0
            DRCANY(2) =  0.0D0
            DRCANZ(2) = 90.0D0
            DRCANX(3) = 90.0D0
            DRCANY(3) = 90.0D0
            DRCANZ(3) = 180.0D0

******   INITIALIZE   ******
*       OMEGA AND OMEGA DOT
160        DO 170 I = 1,3
              W1(I)     = 0.0D0
              W2(I)     = 0.0D0
              W3(I)     = 0.0D0
              WDX(I)    = 0.0D0
              WDY(I)    = 0.0D0
              WDZ(I)    = 0.0D0
170        CONTINUE
              W3(1) = 0.158D0
              W3IC  = W3(1)

           THZ = 0.0D0
```

```
*      INITIALIZE MATRIX A AND B TO ZERO
           DO 180 I = 1,27
              DO 175 J = 1,27
                 MATA(I,J) = 0.0D0
175           CONTINUE
               MATB(I) = 0.0D0
180        CONTINUE

*      CONSTANT TO CONVERT STRIP DATA TO DELTA P
         RDVTOV  = 0.2D0
         RAREA   = 0.00312D0
         RLA     = 0.0597/2.0
         RVTOP   = 10.0D0
         RLBTON  = 4.448D0
         RINTOM  = 39.37D0
         RCNFAC  = RDVTOV*RVTOP*RAREA*RLA*RLBTON*RINTOM**2

******    CALCULATIONS    ******

*      WEIGHTS (NEWTONS)
185        WG1 = M1*G
           WG2 = M2*G
           WG3 = M3*G

*      COMPUTE THE LENGTH FROM THE INBOARD JOINT TO COG
           LNCOG1 = DSQRT ( LCOGX(1)*LCOGX(1) + LCOGY(1)*LCOGY(1) +...
                            LCOGZ(1)*LCOGZ(1) )
           LX2 = LCOGX(2) - JX1
           LY2 = LCOGY(2) - JY1
           LZ2 = LCOGZ(2) - JZ1
           LNCOG2 = DSQRT ( LX2*LX2 + LY2*LY2 + LZ2*LZ2 )
           LX3 = LCOGX(3) - JX2
           LY3 = LCOGY(3) - JY2
           LZ3 = LCOGZ(3) - JZ2
           LNCOG3 = DSQRT ( LX3*LX3 + LY3*LY3 + LZ3*LZ3 )
           V3IC = LNCOG3 * W3(1)

*      L(3,2)
       L(3,2) = (13.5 * RCNFAC) / ( MASS(3,2) * G)  - LNCOG3

*      COMPUTE INITIAL INERTIAS BASED ON POINT MASSES
*               IN GLOBAL COORDINATES
190        DO 225 I = 1,3
              RX(I,1) = -L(I,1) * DRCSX(I)
              RX(I,2) =  L(I,2) * DRCSX(I)
              RY(I,1) = -L(I,1) * DRCSY(I)
              RY(I,2) =  L(I,2) * DRCSY(I)
              RZ(I,1) = -L(I,1) * DRCSZ(I)
              RZ(I,2) =  L(I,2) * DRCSZ(I)
200    IXX(I,1) = MASS(I,1) * ((RY(I,1) * RY(I,1)) + (RZ(I,1) * RZ(I,1)))
       IXX(I,2) = MASS(I,2) * ((RY(I,2) * RY(I,2)) + (RZ(I,2) * RZ(I,2)))
       IXXT(I)  = IXX(I,1) + IXX(I,2)
       IYY(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RZ(I,1) * RZ(I,1)))
       IYY(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RZ(I,2) * RZ(I,2)))
       IYYT(I)  = IYY(I,1) + IYY(I,2)
```

```
      IZZ(I,1) = MASS(I,1) * ((RX(I,1) * RX(I,1)) + (RY(I,1) * RY(I,1)))
      IZZ(I,2) = MASS(I,2) * ((RX(I,2) * RX(I,2)) + (RY(I,2) * RY(I,2)))
      IZZT(I)  = IZZ(I,1) + IZZ(I,2)
205   IXY(I,1) = MASS(I,1) * RX(I,1) * RY(I,1)
      IXY(I,2) = MASS(I,2) * RX(I,2) * RY(I,2)
      IXYT(I)  = IXY(I,1) + IXY(I,2)
      IXZ(I,1) = MASS(I,1) * RZ(I,1) * RX(I,1)
      IXZ(I,2) = MASS(I,2) * RZ(I,2) * RX(I,2)
      IXZT(I)  = IXZ(I,1) + IXZ(I,2)
      IYZ(I,1) = MASS(I,1) * RY(I,1) * RZ(I,1)
      IYZ(I,2) = MASS(I,2) * RY(I,2) * RZ(I,2)
      IYZT(I)  = IYZ(I,1) + IYZ(I,2)
      IF (IXXT(I) .LE. .10) THEN
      IXXT(I)  = .10
      ELSE
      IXXT(I) = IXXT(I)
      END IF
      IF (IYYT(I) .LE. .01) THEN
      IYYT(I)  = .01
      ELSE
      IYYT(I)  = IYYT(I)
      END IF
      IF (IZZT(I) .LE. .01) THEN
      IZZT(I)  = .01
      ELSE
      IZZT(I)  = IZZT(I)
      END IF
           IMAT(I,1,1) =  IXXT(I)
           IMAT(I,1,2) =  IXYT(I)
           IMAT(I,1,3) =  IXZT(I)
           IMAT(I,2,1) = -IXYT(I)
           IMAT(I,2,2) =  IYYT(I)
           IMAT(I,2,3) =  IYZT(I)
           IMAT(I,3,1) = -IXZT(I)
           IMAT(I,3,2) = -IYZT(I)
           IMAT(I,3,3) =  IZZT(I)
225        CONTINUE

*       DUE TO LINK 1 CONSTRAINTS LINK 1 INERTIAS ARE CONSTANT
          IXXT(1) = IMAT(1,1,1)
          IXYT(1) = IMAT(1,1,2)
          IXZT(1) = IMAT(1,1,3)
          IYYT(1) = IMAT(1,2,2)
          IYZT(1) = IMAT(1,2,3)
          IZZT(1) = IMAT(1,3,3)

*     TRANSFORM THE INITIAL INERTIAS TO LOCAL COORDINATED

          DO 9 I = 2, 3
            TMAT(2,1) = -DCOS ( THZ )
            TMAT(2,2) = -DSIN ( THZ )
            TMAT(2,3) = 0.0D0
            TMAT(3,1) = DRCSX(I)
            TMAT(3,2) = DRCSY(I)
            TMAT(3,3) = DRCSZ(I)
```

```
            VECTA(1)  = TMAT(2,1)
            VECTA(2)  = TMAT(2,2)
            VECTA(3)  = TMAT(2,3)
            VECTB(1)  = TMAT(3,1)
            VECTB(2)  = TMAT(3,2)
            VECTB(3)  = TMAT(3,3)
            CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
            TMAT(1,1) = MI1
            TMAT(1,2) = MJ1
            TMAT(1,3) = MK1
            TMATTR(1,1) = TMAT(1,1)
            TMATTR(1,2) = TMAT(2,1)
            TMATTR(1,3) = TMAT(3,1)
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)

            DO 5 I1 = 1,3
            DO 5 J  = 1,3
               TEMP = 0.0D0
              DO 1  K  = 1,3
                 TEMP = TMAT(I1,K) * IMAT(I,K,J) + TEMP
1             CONTINUE
               MATTMP(I1,J) = TEMP
5           CONTINUE

            DO 8   I1 = 1,3
            DO 8   J  = 1,3
               TEMP = 0.0D0
              DO 7   K  = 1,3
                 TEMP = MATTMP(I1,K) * TMATTR(K,J) + TEMP
7             CONTINUE
             LIMAT(I,I1,J) = TEMP
8           CONTINUE
9         CONTINUE

DERIVATIVE
NOSORT
230         CALL ERRSET (208,256,-1,1,1)
            CALL UERSET(LEVELQ,LEVLDQ)


*     INITIALIZE MATRIX A AND B TO ZERO
            DO 240 I = 1,27
               DO 235 J = 1,27
                  MATA(I,J) = 0.0D0
235           CONTINUE
               MATB(I) = 0.0D0
240         CONTINUE

*      INPUT JOINT EQUATIONS
```

```
*       JOINT ZERO  EQUATIONS
*       W1 X (W1 X RB/G1)
              RBG1(1) = JX0 - LCOGX(1)
              RBG1(2) = JY0 - LCOGY(1)
              RBG1(3) = JZ0 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD(VECTA0,RBG1,MIC0,MJC0,MKC0)
              VECTB0(1) = MIC0
              VECTB0(2) = MJC0
              VECTB0(3) = MKC0
           CALL CPROD(VECTA0,VECTB0,MIC0,MJC0,MKC0)

*       INPUT JOINT EQUATIONS
*       JOINT ZERO  EQUATIONS
*        W1 X (W1 X RB/G1)
              RBG1(1)  = JX0 - LCOGX(1)
              RBG1(2)  = JY0 - LCOGY(1)
              RBG1(3)  = JZ0 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD(VECTA,RBG1,MIC0,MJC0,MKC0)
              VECTB(1) = MIC0
              VECTB(2) = MJC0
              VECTB(3) = MKC0
           CALL CPROD(VECTA,VECTB,MIC0,MJC0,MKC0)
*       W1 X (W1 X RA/G1)
              RAG1(1) = JX1 - LCOGX(1)
              RAG1(2) = JY1 - LCOGY(1)
              RAG1(3) = JZ1 - LCOGZ(1)
              VECTA(1) = W1(1)
              VECTA(2) = W1(2)
              VECTA(3) = W1(3)
           CALL CPROD (VECTA,RAG1,MIC1,MJC1,MKC1)
              VECTB(1) = MIC1
              VECTB(2) = MJC1
              VECTB(3) = MKC1
           CALL CPROD (VECTA,VECTB,MIC1,MJC1,MKC1)
*       W2 X (W2 X RB/G2)
              RBG2(1) = JX1 - LCOGX(2)
              RBG2(2) = JY1 - LCOGY(2)
              RBG2(3) = JZ1 - LCOGZ(2)
              VECTA(1) = W2(1)
              VECTA(2) = W2(2)
              VECTA(3) = W2(3)
           CALL CPROD (VECTA,RBG2,MIC2,MJC2,MKC2)
              VECTB(1) = MIC2
              VECTB(2) = MJC2
              VECTB(3) = MKC2
           CALL CPROD (VECTA,VECTB,MIC2,MJC2,MKC2)
*       W2 X (W2 X RA/G2)
              RAG2(1) = JX2 - LCOGX(2)
              RAG2(2) = JY2 - LCOGY(2)
              RAG2(3) = JZ2 - LCOGZ(2)
```

```
            VECTA(1) = W2(1)
            VECTA(2) = W2(2)
            VECTA(3) = W2(3)
          CALL CPROD (VECTA,RAG2,MIC3,MJC3,MKC3)
            VECTB(1) = MIC3
            VECTB(2) = MJC3
            VECTB(3) = MKC3
          CALL CPROD(VECTA,VECTB,MIC3,MJC3,MKC3)
*       W3 X (W3 X RB/G3)
            RBG3(1) = JX2 - LCOGX(3)
            RBG3(2) = JY2 - LCOGY(3)
            RBG3(3) = JZ2 - LCOGZ(3)
            VECTA(1) = W3(1)
            VECTA(2) = W3(2)
            VECTA(3) = W3(3)
          CALL CPROD (VECTA,RBG3,MIC4,MJC4,MKC4)
            VECTB(1) = MIC4
            VECTB(2) = MJC4
            VECTB(3) = MKC4
          CALL CPROD (VECTA,VECTB,MIC4,MJC4,MKC4)

*       INERTIA TRANSFORMATION
          DO 390 I = 2, 3
            TMAT(2,1) = -DCOS ( THZ )
            TMAT(2,2) = -DSIN ( THZ )
            TMAT(2,3) = 0.0D0
            TMAT(3,1) = DRCSX(I)
            TMAT(3,2) = DRCSY(I)
            TMAT(3,3) = DRCSZ(I)

            VECTA(1)  = TMAT(2,1)
            VECTA(2)  = TMAT(2,2)
            VECTA(3)  = TMAT(2,3)
            VECTB(1)  = TMAT(3,1)
            VECTB(2)  = TMAT(3,2)
            VECTB(3)  = TMAT(3,3)
            CALL CPROD (VECTA,VECTB,MI1,MJ1,MK1)
            TMAT(1,1) = MI1
            TMAT(1,2) = MJ1
            TMAT(1,3) = MK1
            TMATTR(1,1) = TMAT(1,1)
            TMATTR(1,2) = TMAT(2,1)
            TMATTR(1,3) = TMAT(3,1)
            TMATTR(2,1) = TMAT(1,2)
            TMATTR(2,2) = TMAT(2,2)
            TMATTR(2,3) = TMAT(3,2)
            TMATTR(3,1) = TMAT(1,3)
            TMATTR(3,2) = TMAT(2,3)
            TMATTR(3,3) = TMAT(3,3)

            DO 375 I1 = 1,3
            DO 375 J  = 1,3
               TEMP = 0.0D0
              DO 370  K  = 1,3
                 TEMP = TMATTR(I1,K) * LIMAT(I,K,J) + TEMP
370           CONTINUE
```

```
               MATTMP(I1,J) = TEMP
375         CONTINUE

            DO 380 I1 = 1,3
            DO 380 J  = 1,3
               TEMP = 0.0D0
              DO 378 K  = 1,3
                 TEMP = MATTMP(I1,K) * TMAT(K,J) + TEMP
378           CONTINUE
             NIMAT(I1,J) = TEMP
380         CONTINUE

             IXXT(I) = NIMAT(1,1)
             IXYT(I) = NIMAT(1,2)
             IXZT(I) = NIMAT(1,3)
             IYYT(I) = NIMAT(2,2)
             IYZT(I) = NIMAT(2,3)
             IZZT(I) = NIMAT(3,3)

390     CONTINUE

******  ENTER CONSTANTS INTO MATRIX A  ******

**      LINK ONE
*       SUM OF FORCES
*       X DIRECTION
395       MATA(1,1)  =  1.0D0
          MATA(1,4)  =  M1
          MATA(1,10) = -1.0D0
          MATB(1)    =  0.0D0
*       Y DIRECTION
          MATA(2,2)  =  1.0D0
          MATA(2,5)  =  M1
          MATA(2,11) = -1.0D0
          MATB(2)    =  0.0D0
*       Z DIRECTION
          MATA(3,3)  =  1.0D0
          MATA(3,6)  =  M1
          MATA(3,12) = -1.0D0
          MATB(3)    = -WG1
*       EQUATIONS AT JOINT ZERO
*       X DIRECTION
          MATA(4,4) =  1.0D0
          MATA(4,8) =  RBG1(3)
          MATA(4,9) = -RBG1(2)
          MATB(4)   =  AX0 - MIC0
*       Y DIRECTION
          MATA(5,5) =  1.0D0
          MATA(5,7) = -RBG1(3)
          MATA(5,9) =  RBG1(1)
          MATB(5)   =  AY0 - MJC0
*       Z DIRECTION
          MATA(6,6) =  1.0D0
          MATA(6,7) =  RBG1(2)
          MATA(6,8) = -RBG1(1)
```

```
          MATB(6)   =  AZ0 - MKC0
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(7,2)  =  RBG1(3)
          MATA(7,3)  = -RBG1(2)
          MATA(7,7)  = -IXXT(1)
          MATA(7,8)  =  IXYT(1)
          MATA(7,9)  =  IXZT(1)
          MATA(7,11) = -RAG1(3)
          MATA(7,12) =  RAG1(2)
          MATB(7)    =  T1X - T0X
*       Y DIRECTION
          MATA(8,1)  = -RBG1(3)
          MATA(8,3)  =  RBG1(1)
          MATA(8,7)  =  IXYT(1)
          MATA(8,8)  = -IYYT(1)
          MATA(8,9)  =  IYZT(1)
          MATA(8,10) =  RAG1(3)
          MATA(8,12) = -RAG1(1)
          MATB(8)    =  T1Y - T0Y
*       Z DIRECTION
          MATA(9,1)  =  RBG1(2)
          MATA(9,2)  = -RBG1(1)
          MATA(9,7)  =  IXZT(1)
          MATA(9,8)  =  IYZT(1)
          MATA(9,9)  = -IZZT(1)
          MATA(9,10) = -RAG1(2)
          MATA(9,11) =  RAG1(1)
          MATB(9)    =  T1Z - T0Z

**      LINK TWO
*       SUM OF FORCES
*       X DIRECTION
460       MATA(10,10) =  1.0D0
          MATA(10,13) =  M2
          MATA(10,19) = -1.0D0
          MATB(10)    =  0.0D0
*       Y DIRECTION
          MATA(11,11) =  1.0D0
          MATA(11,14) =  M2
          MATA(11,20) = -1.0D0
          MATB(11)    =  0.0D0
*       Z DIRECTION
          MATA(12,12) =  1.0D0
          MATA(12,15) =  M2
          MATA(12,21) = -1.0D0
          MATB(12)    = -WG2
*       EQUATIONS AT JOINT ONE
*       X DIRECTION
          MATA(13,4)  = -1.0D0
          MATA(13,8)  = -RAG1(3)
          MATA(13,9)  =  RAG1(2)
          MATA(13,13) =  1.0D0
          MATA(13,17) =  RBG2(3)
          MATA(13,18) = -RBG2(2)
```

```
          MATB(13)    =  MIC1 - MIC2
*       Y DIRECTION
          MATA(14,5)  = -1.0D0
          MATA(14,7)  =  RAG1(3)
          MATA(14,9)  = -RAG1(1)
          MATA(14,14) =  1.0D0
          MATA(14,16) = -RBG2(3)
          MATA(14,18) =  RBG2(1)
          MATB(14)    =  MJC1 - MJC2
*       Z DIRECTION
          MATA(15,6)  = -1.0D0
          MATA(15,7)  = -RAG1(2)
          MATA(15,8)  =  RAG1(1)
          MATA(15,15) =  1.0D0
          MATA(15,16) =  RBG2(2)
          MATA(15,17) = -RBG2(1)
          MATB(15)    =  MKC1 - MKC2
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(16,11) =  RBG2(3)
          MATA(16,12) = -RBG2(2)
          MATA(16,16) = -IXXT(2)
          MATA(16,17) =  IXYT(2)
          MATA(16,18) =  IXZT(2)
          MATA(16,20) = -RAG2(3)
          MATA(16,21) =  RAG2(2)
          MATB(16)    =  T2X - T1X
*       Y DIRECTION
          MATA(17,10) = -RBG2(3)
          MATA(17,12) =  RBG2(1)
          MATA(17,16) =  IXYT(2)
          MATA(17,17) = -IYYT(2)
          MATA(17,18) =  IYZT(2)
          MATA(17,19) =  RAG2(3)
          MATA(17,21) = -RAG2(1)
          MATB(17)    =  T2Y - T1Y
*       Z DIRECTION
          MATA(18,10) =  RBG2(2)
          MATA(18,11) = -RBG2(1)
          MATA(18,16) =  IXZT(2)
          MATA(18,17) =  IYZT(2)
          MATA(18,18) = -IZZT(2)
          MATA(18,19) = -RAG2(2)
          MATA(18,20) =  RAG2(1)
          MATB(18)    =  T2Z - T1Z

**      LINK THREE
*       SUM OF FORCES
*       X DIRECTION
530       MATA(19,19) =  1.0D0
          MATA(19,22) =  M3
          MATB(19)    =  0.0D0
*       Y DIRECTION
          MATA(20,20) =  1.0D0
          MATA(20,23) =  M3
```

```
          MATB(20)    =  0.0D0
*       Z DIRECTION
          MATA(21,21) =  1.0D0
          MATA(21,24) =  M3
          MATB(21) = -WG3
*       EQUATIONS AT JOINT TWO
*       X DIRECTION
          MATA(22,13) = -1.0D0
          MATA(22,17) = -RAG2(3)
          MATA(22,18) =  RAG2(2)
          MATA(22,22) =  1.0D0
          MATA(22,26) =  RBG3(3)
          MATA(22,27) = -RBG3(2)
          MATB(22)    =  MIC3 - MIC4
*       Y DIRECTION
          MATA(23,14) = -1.0D0
          MATA(23,16) =  RAG2(3)
          MATA(23,18) = -RAG2(1)
          MATA(23,23) =  1.0D0
          MATA(23,25) = -RBG3(3)
          MATA(23,27) =  RBG3(1)
          MATB(23)    =  MJC3 - MJC4
*       Z DIRECTION
          MATA(24,15) = -1.0D0
          MATA(24,16) = -RAG2(2)
          MATA(24,17) =  RAG2(1)
          MATA(24,24) =  1.0D0
          MATA(24,25) =  RBG3(2)
          MATA(24,26) = -RBG3(1)
          MATB(24)    =  MKC3 - MKC4
*       SUM OF MOMENTS EQUATIONS
*       X DIRECTION
          MATA(25,20) =  RBG3(3)
          MATA(25,21) = -RBG3(2)
          MATA(25,25) = -IXXT(3)
          MATA(25,26) =  IXYT(3)
          MATA(25,27) =  IXZT(3)
          MATB(25)    = -T2X
*       Y DIRECTION
          MATA(26,19) = -RBG3(3)
          MATA(26,21) =  RBG3(1)
          MATA(26,25) =  IXYT(3)
          MATA(26,26) = -IYYT(3)
          MATA(26,27) =  IYZT(3)
          MATB(26)    = -T2Y
*       Z DIRECTION
          MATA(27,19) =  RBG3(2)
          MATA(27,20) = -RBG3(1)
          MATA(27,25) =  IXZT(3)
          MATA(27,26) =  IYZT(3)
          MATA(27,27) = -IZZT(3)
          MATB(27)    = -T2Z

******    FIRST LINK IS CONSTRAINED NOT TO ROTATE ******
          DO 600 I = 4,9
            DO 595 J = 1,27
```

```
            MATA(I,J) = 0.0D0
            MATA(J,I) = 0.0D0
595        CONTINUE
            MATB(I) = 0.0D0
600      CONTINUE
            MATA(4,4) = 1.0D0
            MATA(5,5) = 1.0D0
            MATA(6,6) = 1.0D0
            MATA(7,7) = 1.0D0
            MATA(8,8) = 1.0D0
            MATA(9,9) = 1.0D0
         DO 602 I = 13,17
           DO 597 J = 1,27
            MATA(I,J) = 0.0D0
            MATA(J,I) = 0.0D0
597        CONTINUE
            MATB(I) = 0.0D0
602      CONTINUE
            MATA(13,13) = 1.0D0
            MATA(14,14) = 1.0D0
            MATA(15,15) = 1.0D0
            MATA(16,16) = 1.0D0
            MATA(17,17) = 1.0D0
605      DO 610 J = 1,27
           MATA(18,J) = 0.0D0
           MATA(27,J) = 0.0D0
610      CONTINUE

           MATA(18,9)  = -1.0D0
           MATA(18,18) =  1.0D0
           MATB(18)    =  0.0D0
           MATA(27,9)  = -1.0D0
           MATA(27,27) =  1.0D0
           MATB(27)    =  0.0D0


*       CALL EQUATION SOLVER PROGRAM FROM IMSL
620        CALL LEQT2F(MATA,M,N,IA,MATB,IDGT,WKAREA,IER)
           IF (IER.EQ.0) THEN
           GOTO 640
           ELSE
           WRITE (*,624) IER
624        FORMAT (I5)
           DO 635  I = 1, 27
           WRITE (*,627) I
627        FORMAT (I7)
           DO 631  J = 1, 27, 3
           WRITE (*,630) J,MATA(I,J),J+1,MATA(I,J+1),J+2,MATA(I,J+2)
630        FORMAT (I5,F13.5,I5,F13.5,I5,F13.5)
631        CONTINUE
           WRITE (*,633) I,MATB(I)
633        FORMAT (I3,F15.5)
635        CONTINUE
           CALL ENDJOB
           END IF
```

```
***     FIND LCOGX,LCOGY,LCOGZ,THETA VALUES,WX,WY,WZ
***     MODIFIED BY R.M.VERBOS

*       JOINT ZERO
640        FX0 = MATB(1)
           FY0 = MATB(2)
           FZ0 = MATB(3)

*       LINK ONE
*       SINCE LINK1 IS CONSTRAIN TO ROTATE AROUND THE Z ONLY
           AX1 = 0.0D0
           AY1 = 0.0D0
           AZ1 = 0.0D0
*660       AX1        = MATB(4)
*          VELX1      = INTGRL(0.,AX1)
*          LCOGX1     = INTGRL(X1,VELX1)
*          LCOGX(1)   = LCOGX1
*          AY1        = MATB(5)
*          VELY1      = INTGRL(0.,AY1)
*          LCOGY1     = INTGRL(Y1,VELY1)
*          LCOGY(1)   = LCOGY1
*          AZ1        = MATB(6)
*          VELZ1      = INTGRL(0.,AZ1)
*          LCOGZ1     = INTGRL(Z1,VELZ1)
*          LCOGZ(1)   = LCOGZ1
         WD1X       = MATB(7)
         W1X        = INTGRL(0.,WD1X)
         WDX(1)     = WD1X
         W1(1)      = W1X
         WD1Y       = MATB(8)
         W1Y        = INTGRL(0.,WD1Y)
         WDY(1)     = WD1Y
         W1(2)      = W1Y
         WD1Z       = MATB(9)
         W1Z        = INTGRL(0.,WD1Z)
         WDZ(1)     = WD1Z
         W1(3)      = W1Z
***      ADDED BY R.M.VERBOS
685      THZ        = INTGRL(0.,W1Z)
         THZANG     = THZ * RADEG
         COSTHZ     = DCOS(THZ)
         SINTHZ     = DSIN(THZ)

*   IF THE 1ST LINK IS CONSTRAIN TO ROTATE IN THE Z DIRECTION ONLY
*   THE DIRECTION COSINE AND DIRECTION COSINE  ANGLES ARE CONSTANT
*   AND DO NOT NEED TO BE CALCULATED
*
*   CALC DIRECTIONAL COSINES FOR LINK ONE
*
*695       LNCOG1 = DSQRT ( LCOGX1*LCOGX1 + LCOGY1*LCOGY1 +...
*                           LCOGZ1*LCOGZ1 )
*700       DRCSX(1)  = LCOGX1 / LNCOG1
*          DRCSY(1)  = LCOGY1 / LNCOG1
*          DRCSZ(1)  = LCOGZ1 / LNCOG1
*         AMP = DSQRT(DRCSX(1)*DRCSX(1)+DRCSY(1)*DRCSY(1)+...
*                     DRCSZ(1)*DRCSZ(1))
```

```
*          DRCSX(1) = DRCSX(1)/AMP
*          DRCSY(1) = DRCSY(1)/AMP
*          DRCSZ(1) = DRCSZ(1)/AMP
*720       DRCANX(1) = DACOS(DRCSX(1)) * RADEG
*          DRCANY(1) = DACOS(DRCSY(1)) * RADEG
*          DRCANZ(1) = DACOS(DRCSZ(1)) * RADEG
*
**       JOINT LOCATION
*730       JX1  =  LINKL1 * DRCSX(1)
*          JY1  =  LINKL1 * DRCSY(1)
*          JZ1  =  LINKL1 * DRCSZ(1)

*        JOINT ONE
735        FX1 = MATB(10)
           FY1 = MATB(11)
           FZ1 = MATB(12)

           AX2 = 0.0D0
           AY2 = 0.0D0
           AZ2 = 0.0D0

**       LINK TWO
740        AX2        = MATB(13)
*          VELX2      = INTGRL(0.,AX2)
*          LCOGX2     = INTGRL(X2,VELX2)
*          LCOGX(2)   = LCOGX2
*          AY2        = MATB(14)
*          VELY2      = INTGRL(0.,AY2)
*          LCOGY2     = INTGRL(Y2,VELY2)
*          LCOGY(2)   = LCOGY2
*          AZ2        = MATB(15)
*          VELZ2      = INTGRL(0.,AZ2)
*          LCOGZ2     = INTGRL(Z2,VELZ2)
*          LCOGZ(2)   = LCOGZ2
           WD2X       = MATB(16)
           W2X        = INTGRL(0.,WD2X)
           WDX(2)     = WD2X
           W2(1)      = W2X
           WD2Y       = MATB(17)
           W2Y        = INTGRL(0.,WD2Y)
           WDY(2)     = WD2Y
           W2(2)      = W2Y
           WD2Z       = MATB(18)
           W2Z        = INTGRL(0.,WD2Z)
           WDZ(2)     = WD2Z
           W2(3)      = W2Z

*        GET THE DIRECTION COSINES FOR THE LINK TWO
*          DRCSX(2)   = (LCOGX2 - JX1) / LNCOG2
*          DRCSY(2)   = (LCOGY2 - JY1) / LNCOG2
*          DRCSZ(2)   = (LCOGZ2 - JZ1) / LNCOG2
*90        AMP = DSQRT(DRCSX(2)*DRCSX(2)+DRCSY(2)*DRCSY(2)+...
*                      DRCSZ(2)*DRCSZ(2))
*          DRCSX(2) = DRCSX(2)/AMP
*          DRCSY(2) = DRCSY(2)/AMP
*          DRCSZ(2) = DRCSZ(2)/AMP
```

```
*         DRCANX(2) = DACOS(DRCSX(2)) * RADEG
*         DRCANY(2) = DACOS(DRCSY(2)) * RADEG
*         DRCANZ(2) = DACOS(DRCSZ(2)) * RADEG

*      JOINT LOCATION
800       JX2 = JX1 + LINKL2 * DRCSX(2)
          JY2 = JY1 + LINKL2 * DRCSY(2)
          JZ2 = JZ1 + LINKL2 * DRCSZ(2)

*      JOINT TWO
805       FX2 = MATB(19)
          FY2 = MATB(20)
          FZ2 = MATB(21)

**      LINK THREE
812       AX3        = MATB(22)
          VELX3      = INTGRL(0.,AX3)
          LCOGX3     = INTGRL(X3,VELX3)
          LCOGX(3)   = LCOGX3
          AY3        = MATB(23)
          VELY3      = INTGRL(V3IC,AY3)
          LCOGY3     = INTGRL(Y3,VELY3)
          LCOGY(3)   = LCOGY3
          AZ3        = MATB(24)
          VELZ3      = INTGRL(0.,AZ3)
          LCOGZ3     = INTGRL(Z3,VELZ3)
          LCOGZ(3)   = LCOGZ3
          WD3X       = MATB(25)
          W3X        = INTGRL(W3IC,WD3X)
          WDX(3)     = WD3X
          W3(1)      = W3X
          WD3Y       = MATB(26)
          W3Y        = INTGRL(0.,WD3Y)
          WDY(3)     = WD3Y
          W3(2)      = W3Y
          WD3Z       = MATB(27)
          W3Z        = INTGRL(0.,WD3Z)
          WDZ(3)     = WD3Z
          W3(3)      = W3Z

*  CALC DIRECTIONAL COSINES FOR LINK THREE
845       DRCSX(3) = (LCOGX3 - JX2) / LNCOG3
          DRCSY(3) = (LCOGY3 - JY2) / LNCOG3
          DRCSZ(3) = (LCOGZ3 - JZ2) / LNCOG3
865       AMP = DSQRT(DRCSX(3)*DRCSX(3)+DRCSY(3)*DRCSY(3)+...
                      DRCSZ(3)*DRCSZ(3))
          DRCSX(3) = DRCSX(3)/AMP
          DRCSY(3) = DRCSY(3)/AMP
          DRCSZ(3) = DRCSZ(3)/AMP
          DRCANX(3) = DACOS(DRCSX(3)) * RADEG
          DRCANY(3) = DACOS(DRCSY(3)) * RADEG
          DRCANZ(3) = DACOS(DRCSZ(3)) * RADEG

*   TIP LOCATION
875       TIPX = JX2 + LINKL3 * DRCSX(3)
          TIPY = JY2 + LINKL3 * DRCSY(3)
```

```
          TIPZ = JZ2 + LINKL3 * DRCSZ(3)

*         FIND THE ANGLE BETWEEN THE LIMKS
880       ANG23X = DRCANX(2) - DRCANX(3)
          ANG23Y = DRCANY(2) - DRCANY(3)
          ANG23Z = DRCANZ(2) - DRCANZ(3)
          ANG12X = DRCANX(1) - DRCANX(2)
          ANG12Y = DRCANY(1) - DRCANY(2)
          ANG12Z = DRCANZ(1) - DRCANZ(2)

DYNAMIC
NOSORT
******  INPUT TORQUE AT JOINTS
            T2FNC = (13.5 * SIN (PI / (30.0) * TIME)) * RCNFAC
            T2 = T2FNC
            T2X = T2

END
STOP


FORTRAN

*      SUBROUTINE TO COMPUTE THE CROSS PRODUCT OF TWO VECTORS

          SUBROUTINE CPROD(VECTA,VECTB,MI,MJ,MK)
940           IMPLICIT REAL*8 (A-Z)
              DIMENSION VECTA(3),VECTB(3)
              MI = VECTA(2) * VECTB(3) - VECTA(3) * VECTB(2)
              MJ = VECTA(3) * VECTB(1) - VECTA(1) * VECTB(3)
              MK = VECTA(1) * VECTB(2) - VECTA(2) * VECTB(1)
           RETURN
           END
```

APPENDIX E

BASIC OPERATING INSTRUCTIONS FOR THE NAVAL POSTGRADUATE SCHOOL RIGID MANIPULATOR TEST BED

A. SYSTEM FAMILIARIZATION

Before operating the manipulator test bed, you should ensure you are familiar with its components and basic procedures of operation. If you are already familiar with the components of the manipulator and their set-up, you will need to review this section to operate the hardware.

The test bed consists of four major sub-systems: the Neptune II hydraulically operated manipulator; an IBM PC-AT used to control the manipulator and gather data (at the present time); an IBM PC-XT (not in the system at present); and an Electronic Control Panel used in switching, data gathering, and interfacing.

1. The Manipulator

The manipulator (Figure E-1), being a hydraulic mechanism, is subject to leakage. It is advisable to wear old clothing while in the lab. The operating pressure of the system is around 100 psi and a minor leak may spray anywhere in the lab.

The source of the hydraulic fluid is the pump cabinet against the forward side of the wall in front of the computers. This cabinet contains a pump which is operated from the ECP, a reservoir which is underneath the pump, and an accumulator that is alongside of the pump (Figure E-2).

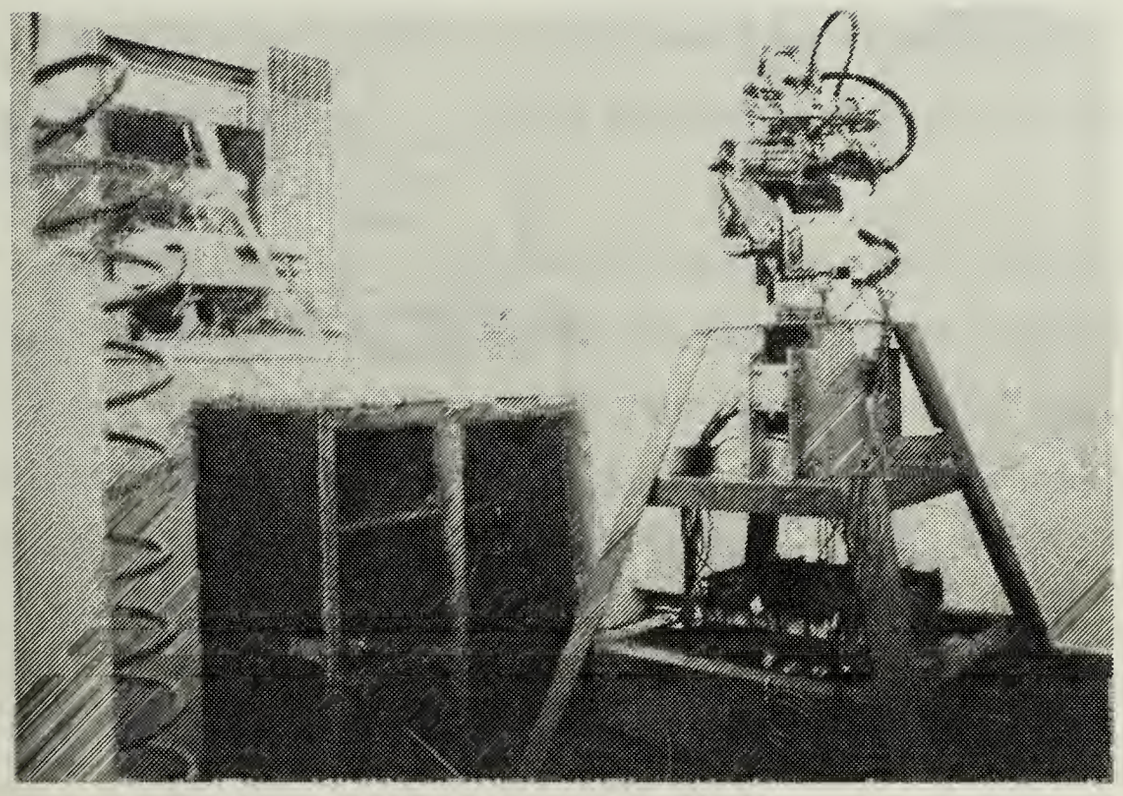


Figure E-1. **Manipulator**

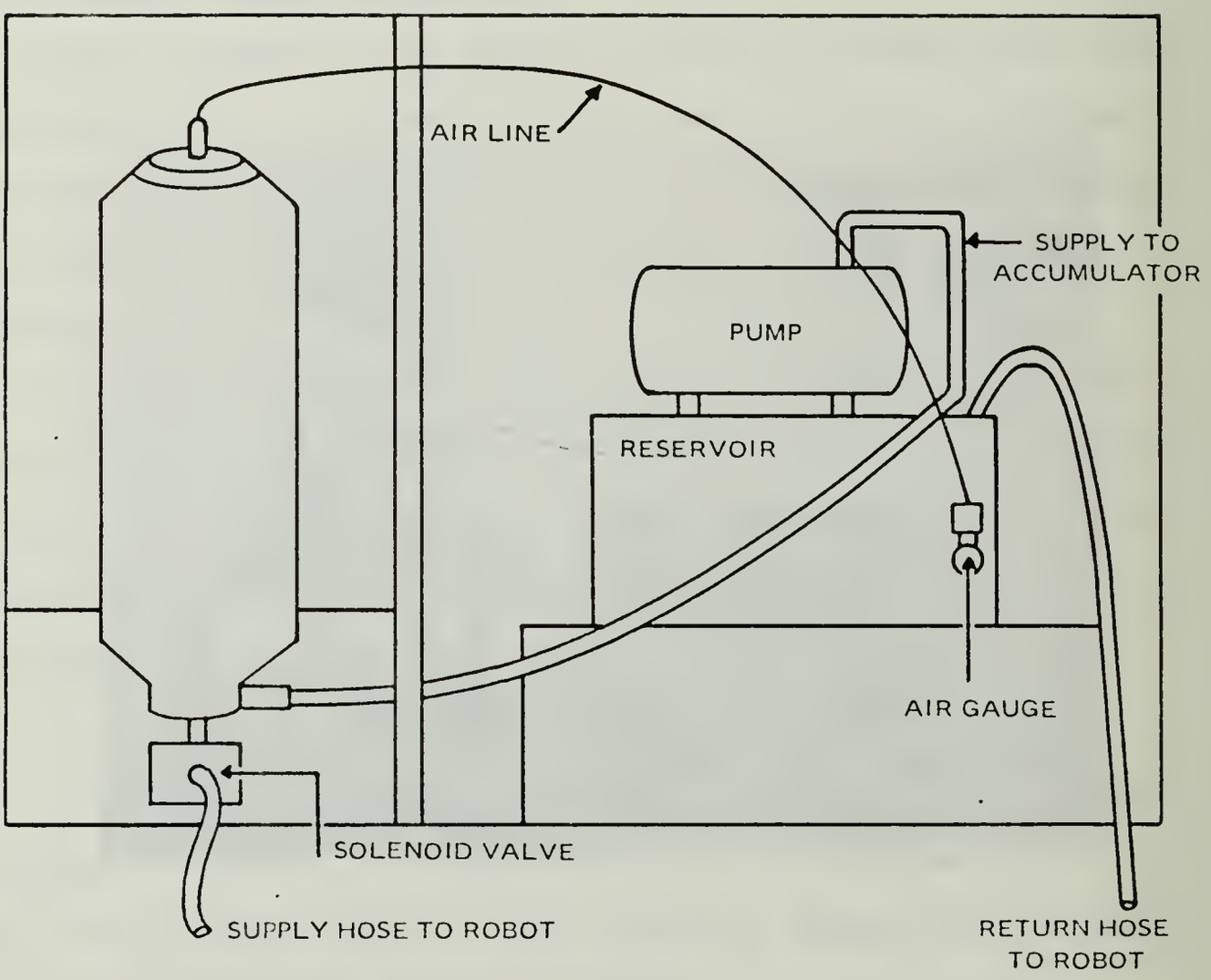


Figure E-2. **Pump Cabinet**

The reservoir has a fill hole on the top. It is necessary to check this before operating the system and ensure that the level is around half when the system is not in operation. You will also notice a pressure gage which reads the pressure in the accumulator. This pressure should be between 80 and 100 psi and may be increased by pressurizing the bladder in the accumulator to a higher pressure.

If you look closely at the accumulator, you will notice a solenoid valve at the outlet. This valve has been installed because of previous problems and will cut off all hydraulic flow to the manipulator when it is shut from the ECP.

Underneath the manipulator itself is a set of Electronic Printed Circuit Boards (EPCB). These boards control the opening and shutting of the solenoid values to the joint pistons (see Ref. 9 for more information). The important thing to know about these boards is that you should avoid getting hydraulic fluid on them.

2. The Computers

At present, the IBM-AT is the only computer hooked up in the system. It is easy to follow because it has been set up with a menu-based operating system. The power to the AT is under the right side of the computer stand. The first thing that will happen when you turn the power on is that the AT will go thru a cold-start boot-up, which may take a minute or two. You will then notice a listing of subdirectories on the screen. Most of these subdirectories are of little use because the files are proprietary in nature and the author has long since transferred away from NPS, but feel free to skim through them.

For the initial running of the manipulator, you will want to select the BATCHES subdirectory. Once you are in this subdirectory, you will select the NEPTROLA file to run the manipulator. This file contains the control program for operating the manipulator in the solenoid mode. The program is a useful starting point for examining the range of the arm motion.

NEPTROLA is a menu-driven set-up that asks various questions to get the system in the proper operating configuration. The actual operation will be discussed in the operation section of this appendix.

3. Electronic Control Panel (ECP)

The ECP is located between the IBM-AT and the IBM-XT and is the heart of the control of the manipulator. The ECP contains several interface boards which should not be tampered with without talking to Tom Cristian of the Mechanical Engineering technical staff. The front of the panel is divided into four sections (Figure E-3). Starting at the top, the first sub-panel consists of two sets of electrical jacks. The first six from the left are connected to the position data of the interface boards and will provide joint 0 thru 5 position output. The last two jacks are for servo operation which is no longer connected (more information on servo operation is available in Ref. 13). The next panel down was for control of the servovalves; again, more information is available in Reference 13. The next panel contains a power switch which selects power to the panel and electrical jacks to

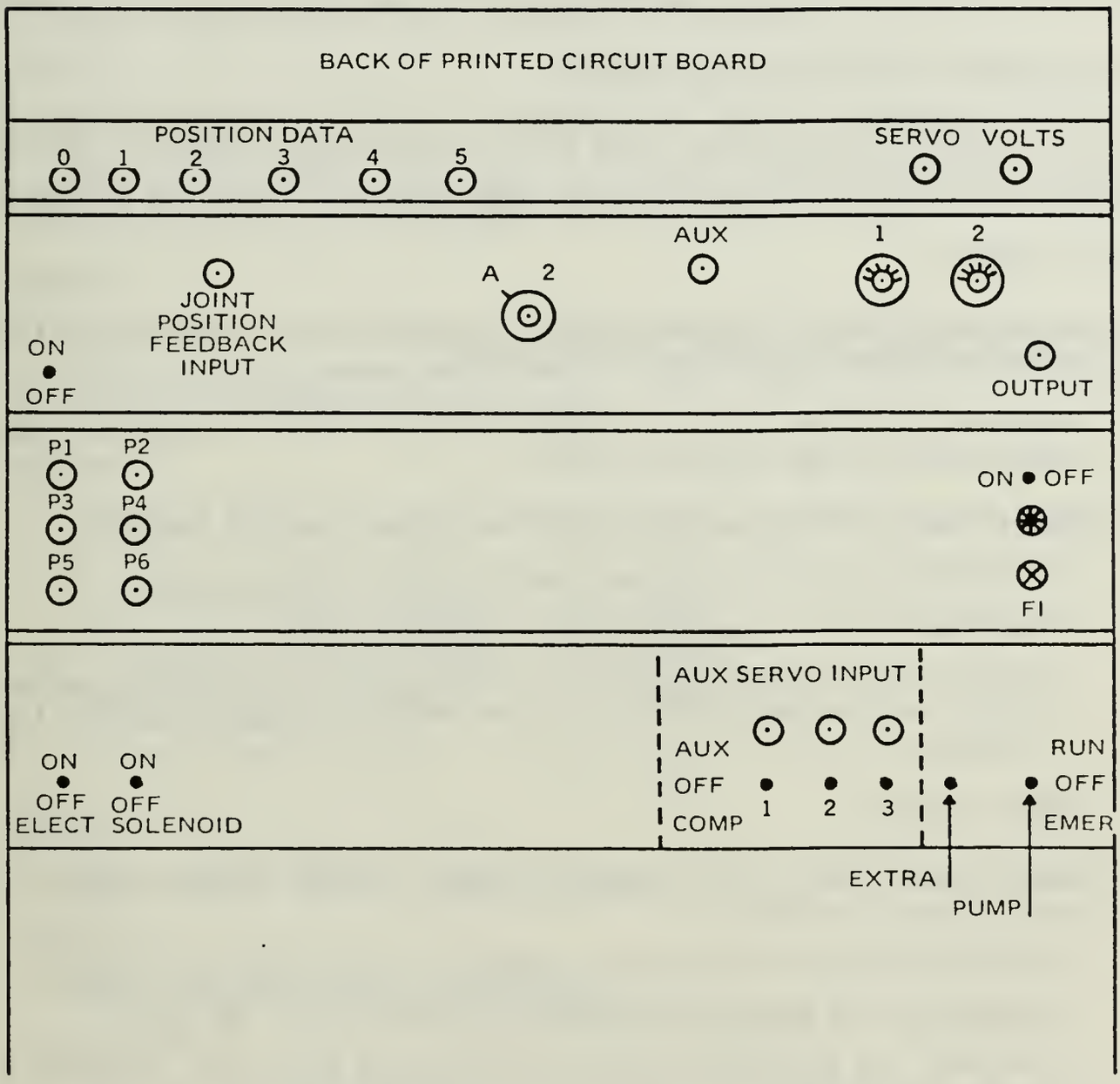


Figure E-3. **Electronic Control Panel (ECP)**

monitor the pressure transducer output for torque calculations. Pressure transducers have been installed on joints 1 and 2. The electrical jacks are wired so that P1 and P2 are connected to joint 1 transducers, P3 and P4 are connected to joint 2 transducers, and P5 and P6 are available for further connections.

The bottom panel is the power panel and consists of seven switches and three electrical jacks. The switches, from left to right, are as follows:

- Electronic Control Power, which supplies the electricity to the interface board at the back of the panel.
- Solenoid Current Switch, which must be "ON" to operate the manipulator in the solenoid mode.
- Servo Master Switches (3), which are three-position switches:
  - (DOWN) is for selecting servovalve control from the AT
  - (MIDDLE) is for operating in the solenoid mode
  - (UP) is for for operating the servovalves from an auxiliary signal which is connected to the three electrical jacks above the switches.
- Extra switch.
- (Rightmost) Pump Power/Solenoid Valve Control Switch. This is a three-position switch:
  - (UP) the pump is in the run mode and controlled by pressure
  - (MIDDLE) the pump is off and the solenoid valve is open
  - (DOWN) the pump is off and the solenoid valve has power to it and is shut (this is an emergency cut-off position).

B. OPERATING THE ARM

1. The Solenoid Mode

Following is a step-by-step procedure for operating the Neptune II manipulator in the solenoid mode:

STEP 1

Check the accumulator to ensure that there is sufficient fluid to operate the manipulator. If the level is low, add a 50/50 mix of Hydro Lube and water until the level is between one-half and three-quarters (DO NOT FILL).

STEP 2

Check the accumulator pressure to ensure it is between 80 and 100 psi. If the pressure is low or high, bleed or add air through the hose connection as necessary.

STEP 3

From the ECP turn the following switches to the position referred to:

- Electronic Power Switch — ON (UP)
- Solenoid Current Switch — ON (UP)
- Servo Master Switches — OFF (MIDDLE)
- Pump Power Switch — ON (UP)

STEP 4

Now turn the IBM-AT computer on.

STEP 5

Select the BATCHES subdirectory.

STEP 6

Select the NEPTROLA program and answer "yes" (Y) to the question whether you want to operate without ADC. This will allow you to operate from the keyboard. The next question will give several options—pick the keyboard option. Follow the given computer menu and vary the joint positions as desired.

STEP 7

To leave the NEPTROLA program, hit ESC, then hit the control and break key at the same time. After this, type "SYSTEM" and you will be returned to the BATCHES subdirectory menu.

2. Servovalve Operation

At present, the servo valves have been removed from the system but the software and wiring are still available. When the servovalves are reinstalled, refer to [Ref. 13] for operating procedures.

LIST OF REFERENCES

1. Fu, K. S., Gonzales, R. C., Lee, C. S. G., *Robotics: Control, Sensing, Vision, and Intelligence*, p. prefix 9, McGraw-Hill Book Co., 1987.

2. Craig, J. J., *Introduction to Robotic Mechanism and Control*, pp. 146–149, Addison-Wesley Co., 1986.

3. Altinok, S., A., *Three Dimensional Non-singular Modeling of Rigid Manipulators*, M.S. Thesis, Naval Postgraduate School, Monterey, California, December 1987.

4. Lewis, D. R., *Modeling of a Low Performance Rigid Revolute Robot Arm*, M.S. Thesis, Naval Postgraduate School, Monterey, California, September 1986.

5. Isaak, S., and Manougian, M. N., *Basic Concepts of Linear Algebra*, pp. 136–161, W. W. Norton and Co., 1976.

6. Frank, A. A., and McGhee, R. B., "Some Considerations Relating to the Design of Autopilots for Legged Vehicles," *J. of Terramechanics*, vol. 6, pp. 23–35, 1969.

7. Greenwood, D. T., *Principles of Dynamics*, pp. 354–370, Prentice-Hall Inc., 1988.

8. Merian, J. L., *Engineering Mechanics Vol. 2, Dynamics*, pp. 469–494, John Wiley and Sons, 1978.

9. Torby, B. J., *Advance Dynamics for Engineers*, pp. 165–167, Holt, Rinehart and Winston, 1984.

10. George Lee, C. S., "Robot Arm Kinematics, Dynamics, and Control," *IEEE* vol. 15, no. 12 (December 1982), pp. 62–80.

11. Meirovitch, L., *Methods of Analytic Dynamics*, pp. 72–79, McGraw-Hill Book Company, 1970.

12. *Neptune II Installation and Maintenance Manual*, Alan-Hayes Corporation, Berkeley Heights, New Jersey (no date).

13. Fancher, C. S., *Manipulator Modeling Sensitivity Studies*, M.S. Thesis, Naval Postgraduate School, Monterey, California, December 1987.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Department Chairman, Code 69 Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4.	Professor D. L. Smith, Code 69Sm Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	10
5.	Professor Nunn, Code 69Nn Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
6.	Professor Chang, Code 69Ck Department of Mechanical Engineering Naval Postgraduate School Monterey, California 93943-5000	1
7.	Professor Robert McGhee, Code 52Mz Department of Computer Science Naval Postgraduate School Monterey, California 93943-5000	1
8.	Chief of Naval Research 800 North Quincy Arlington, Virginia 22217-5000	2

9. Ms. Mary Lacey — 2
Naval Surface Weapons Center
White Oak, Code R402
Robotics Research and Development Center
10908 New Hampshire Avenue
Silver Spring, Maryland 20903-5000

10. Mr. Tom Christian, Code 69 — 1
Department of Mechanical Engineering
Naval Postgraduate School
Monterey, California 93943-5000

11. Mr. S. Altinok — 1
1235 South Main
Salinas, California 93901

12. CDR D. Mahoney, Code 34 — 1
Naval Engineering Curricular Officer
Naval Postgraduate School
Monterey, California 93943-5000

13. LT R. M. Verbos USN — 2
216 Clark Street
Lemoyne, Pennsylvania 17043

Thesis
V394
c.1

Verbos
A three-dimensional
nonsingular simulation
of rigid manipulators.